

PERL

Practical Extraction and Report Language

Programming examples but in
text form as opposed to
graphics

Primarily using ActiveState from:

www.activestate.com/activeperl

and nothing more involved than the Windows notepad or wordpad editors, and double clicking the hDial.pl or vDial.pl resulting programs. ActiveState was chosen as it was simple, installed, and worked first time. THEN after basic experience was gained, then the IDE was used; note that the IDE does not display prompts (for latitude etc) until after the inputs are entered!!.

Strawberry Perl was not used as it did not offer benefits over ActiveState, and seemed to be too Unix oriented.

please check regularly for updates at:

www.illustratingshadows.com/b3/programmingShadows.pdf

which has excellent hints for Perl and the IDE

Simon Wheaton-Smith
www.illustratingshadows.com
January 11, 2014

Download Perl Distributions

Getting started quickly:

 Unix/Linux ✓ Included (may not be latest)	 Mac OS X ✓ Included (may not be latest)	 Windows Windows ↓ Strawberry Perl ↗ ↓ ActiveState Perl ↗
--	--	---

Perl runs on over 100 platforms!

We recommend that you always run the latest stable version, currently 5.18.1. If you're running a version older than 5.8.3, you may find that the latest version of CPAN modules will not work.

TWO CHOICES...

StrawberryPerl
or
ActiveState

I chose ActiveState, it installed first time, had the additional libraries (math, GD, etc) and worked. And the IDE choses also worked with it well.

PERL programming ~ StrawberryPerl ~ I did not use this, I used ActiveState instead

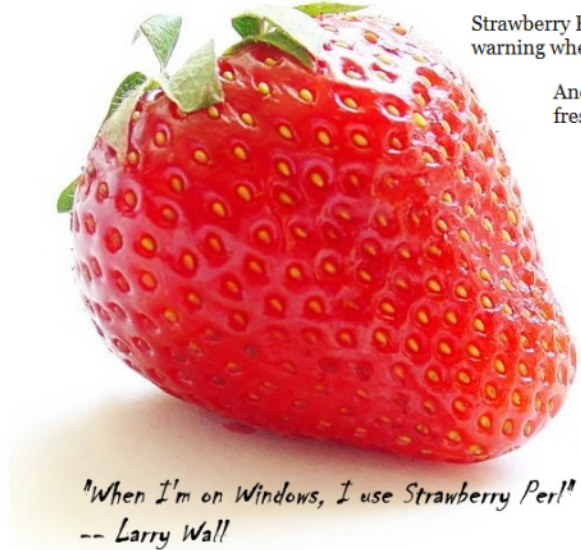
<http://www.perl.org/get.html>

<http://strawberryperl.com/>

#win32 - [Chat Live](#) and get help from other Windows Perl developers

[Need help with a Strawberry Problem?](#)

Signed, Sealed, Delivered, It's Yours!



Strawberry Perl has now been updated, and you get a less annoying warning when installing it, because you can verify that it was built by us.

And it still includes all the bundled database clients and all the fresh CPAN goodness that you expect from Strawberry Perl.

Expected SOON: Perl 5.14.0 - Watch the space below.

Download Strawberry Perl 5.12.3.0 XP/2003/Vista/2008/Win7, English, 40MB, May 2011 (release notes)
Download Strawberry Perl 5.10.1.5 XP/2003/Vista/2008/Win7, English, 34MB, May 2011 (release notes)
Other D: drive, ZIP, Portable, and 64-bit editions

strawberry-perl-5.12.3.0.msi

Strawberry Perl 5.12.3.0 README
=====

What is Strawberry Perl?

- * 'Perl' is a programming language suitable for writing simple scripts as well as complex applications. See <http://perldoc.perl.org/perlintro.html>
- * 'Strawberry Perl' is a perl enviroment for Microsoft Windows containing all you need to run and develop perl applications. It is designed to be as close as possible to perl enviroment on UNIX systems. See <http://strawberryperl.com/>

Installation instructions: (.zip distribution only)

- * If installing this version from a .zip file, you MUST extract it to a directory that does not have spaces in it (referred to as <directory> below) and then run some commands and manually set some environment variables:

```
cd <directory>
perl\bin\perl.exe relocation.pl.bat
perl\bin\perl.exe update_env.pl.bat
```

(You can specify "--nosystem" after update_env.pl.bat to install Strawberry Perl's environment variables for the current user only.)

* If having a fixed installation path does not suit you, go to <http://strawberryperl.com/releases.html> and try "Strawberry Perl Portable Edition"

How to use Strawberry Perl?

* In the command prompt window you can:

1. run any perl script by launching

```
c:\> perl c:\path\to\script.pl
```

2. install additional perl modules (libraries) from <http://www.cpan.org/> by

```
c:\> cpan Module::Name
```

3. run other tools included in Strawberry Perl like: perldoc, gcc, dmake ...

* You'll need a text editor to create perl scripts. One is NOT included with Strawberry Perl. A few options are Padre (which can be installed by running "cpan Padre" from the command prompt for 32-bit versions) and Notepad++ (which is downloadable at notepad-plus.sourceforge.net), which both include syntax highlighting for perl scripts. You can even use Notepad, if you wish.

PERL programming ~ ActiveState ~ I used this, I did not use StrawberryPerl

OR . . .

www.activestate.com/activeperl

ActiveState
Once to Build, Smarter, Safer, Faster™

StackatoDeveloper Tools**Languages**SupportBlog

Sign InStoreContact UsSearch

ActivePerl
The Industry-Standard Perl Distribution

Download Now

[Request More Information](#)

ActivePerl >

ActivePython


ActiveTcl

Compare Editions

Stackato by Language

ActivePerl Business and Enterprise Editions feature our precompiled, supported, quality-assured Perl distribution used by millions of developers around the world for easy Perl installation and quality-assured code. When you're using Perl on production servers or mission-critical applications, ActivePerl Business and Enterprise Editions offer significant time savings over open source Perl for installing, managing, and standardizing your Perl .

Want Perl on the cloud? ActiveState Stackato enables you to deploy Perl applications on a private PaaS. [Learn more.](#)



Business Edition \$399 US
Priced per OS Instance/year

Get a Quote
for support or redistribution rights

Download ActivePerl
Free Community Edition

Which edition is right for you?

Not sure which edition is right for you? Check the [Compare Editions chart](#).

Tested, Timely and Compatible

Save time in your development cycles by starting with a precompiled Perl distribution for out-of-the-box installation and standardization across the operating systems you rely on, including Windows, Linux, Mac OS X, Solaris, AIX, and HP-UX. Develop once and deploy successfully across your production environment.

ActivePerl Business and Enterprise Editions includes:

- > Precompiled Perl distribution for out-of-the-box installation, including ActivePerl 5.16, 5.14, 5.12, 5.10, 5.8, 5.6
- > Core Perl binaries
- > Additional binaries for popular packages/modules through Perl Package Manager (PPM)*
- > Access to PPM Index for easy searching through an additional 10,000 third-party Perl modules
- > Extended documentation

Business Edition licensing is for each production or external-facing server, including virtual servers. Enterprise Edition licensing is on a custom, site-wide basis.

Reduce Risk with Commercially Supported Perl

- > Comply with corporate policy requirements to have supported open source products
- > Full license review including all precompiled third-party Perl modules with assurances to minimize risk (Enterprise Edition only)
- > Protect your organization from legal risk with indemnification coverage (Enterprise Edition only)

Save Time with Support from the Perl Experts

With ActivePerl Business and Enterprise Editions, you get the technical support you need so you can save time and get to market faster.

- > Keep your development issues private with direct access to our Perl support team
- > Count on Perl expertise at your fingertips with Service Level Agreements (SLAs) to meet your business' needs (Enterprise Edition only)

Related Links

[Download ActivePerl Community Edition](#)

[ActivePerl Datasheet \(227kb PDF\)](#)

[IP Infringement Datasheet \(227kb PDF\)](#)

[ActivePerl Documentation](#)

[ActivePerl Community License Agreement](#)



The first end-to-end
enterprise cloud
platform for Python
and Perl applications

[FIND OUT MORE](#)

- Free up precious development time to focus on your core competencies
- Ensure maximum uptime for your business-critical applications

Extended Platform and Version Support

ActivePerl Business and Enterprise Editions extend your ability to develop applications for the following additional operating systems: Solaris, AIX, HP-UX.

Plus, get access to older Perl versions across all platforms:

- Convenient, worry free access to ActivePerl 5.16, 5.14, 5.12, 5.10, 5.8 and 5.6
- Benefit from the ability to test your products against any version release

Support Options

Community Edition	Business Edition
Free	\$999 per OS instance/year
Support through community only	Email support** 2-day response time 1 technical contact
Download	More Info

Enterprise Edition
Custom pricing
Unlimited phone/email support 2-hour response time*** 2 technical contacts
More Info

For more details on which edition is right for you, see the [Compare Editions chart](#).

* Module and platform availability vary by version and operating system. Please see PPM or search [PPM Index](#) for current module availability. Please contact [ActiveState sales department](#) for details on version and platform support for Business and Enterprise Editions.

** Business Edition includes support for build, installation, usage, configuration, and diagnosis dependent on ActiveState's product life cycle.

USING ActiveState from "www.activestate.com/activeperl":-

and documentation from

<http://perldoc.perl.org/Math/Trig.html#TRIGONOMETRIC-FUNCTIONS>

<http://alvinalexander.com/blog/post/perl/reference-page-perl-printf-formatting-format-cheat-sheet>

The following is a text only horizontal dial program

```
# A horizontal sundial with a longitude correction, text only
# www.illustratingshadows.com
# January 4, 2013

use English;
# http://perldoc.perl.org/Math/Trig.html#TRIGONOMETRIC-FUNCTIONS
use Math::Trig;          # needs "use Math::Trig;" as "atan2" is worthless
                        # also TAN is in math.trig, otherwise you must
                        # do sin/cos. But the big deal is TAN vs ATAN2

print "\n*** Illustrating Shadows PERL *** hDial Jan 5, 2014\n";

print "\nEnter a latitude as nn.n    ";
$latitude = readline(*STDIN);

print "\nEnter a longitude offset (+ is west, - is east) as nn.n    ";
$longitude = -readline(*STDIN);

$slat = sin($latitude*2*3.14159/360);

$hr=-6;
while ($hr<7) {          # NOTE: ctrl-C will break a loop in Windows 8

    # following does not need "use Math::Trig;" however, it is needed for ATAN etc
    # $hlaTAN = $slat ;
    # $hlaTAN = $hlaTAN * sin(15*($hr+($longitude*4/60))*2*3.14159/360) ;
    # $hlaTAN = $hlaTAN / cos(15*($hr+($longitude*4/60))*2*3.14159/360) ;

    $hlaTAN = $slat * Math::Trig::tan(15*($hr+($longitude*4/60))*2*3.14159/360);

    $hla    = atan($hlaTAN); # needs "use Math::Trig;" as "atan2" is worthless
    $hla    = $hla*360/(2*3.14159);

    # http://alvinalexander.com/blog/post/perl/reference-page-perl-printf-
    # formatting-format-cheat-sheet
    print "\nhour is: " , $hr+12 , " hour line angle is: ";
    printf("%.2f", $hla);

    $hr = $hr + 1;
}

print "\nbye.\n";
$latitude = readline(*STDIN); # any old input to pause screen
# end #
```

```

C:\Perl64\bin\perl.exe

*** Illustrating Shadows PERL *** hDial Jan 5, 2014
Enter a latitude as nn.n    33.5
Enter a longitude offset (<+ is west, - is east) as nn.n    7.1

hour is: 6 hour line angle is: 77.28
hour is: 7 hour line angle is: -75.89
hour is: 8 hour line angle is: -52.57
hour is: 9 hour line angle is: -35.34
hour is: 10 hour line angle is: -22.66
hour is: 11 hour line angle is: -12.63
hour is: 12 hour line angle is: -3.93
hour is: 13 hour line angle is: 4.38
hour is: 14 hour line angle is: 13.12
hour is: 15 hour line angle is: 23.25
hour is: 16 hour line angle is: 36.12
hour is: 17 hour line angle is: 53.66
hour is: 18 hour line angle is: 77.28
bye.

```

TIME OF DAY		Hour line angles for		Horizontal hour line angles with long corr	
		simple dials		am	pm
am	pm	HORIZONTAL			
12.00	12.00	0.00		3.93	-3.93
11.50	0.50	4.16		8.18	0.22
11.00	1.00	8.41		12.63	4.38
10.50	1.50	12.88		17.41	8.64
10.00	2.00	17.68		22.66	13.12
9.50	2.50	22.95		28.56	17.94
9.00	3.00	28.90		35.34	23.25
8.50	3.50	35.73		43.25	29.24
8.00	4.00	43.71		52.57	36.12
7.50	4.50	53.11		63.48	44.17
7.00	5.00	64.10		75.89	53.66
6.50	5.50	76.58		89.28	64.74
6.00	6.00	90.00		-77.28	77.28
		These have no longitude correction		These have longitude correction	

USING ActiveState from "www.activestate.com/activeperl":-

and documentation as before, the following is a text only horizontal dial program

```
# A vertical sundial with a longitude correction, text only
# www.illustratingshadows.com
# January 5, 2013
# converted from the hDial Perl program, see ... "# change here for h->v Dial"
notes

use English;
# http://perldoc.perl.org/Math/Trig.html#TRIGONOMETRIC-FUNCTIONS
use Math::Trig;          # needs "use Math::Trig;" as "atan2" is worthless
                        # also TAN is in math.trig, otherwise you must
                        # do sin/cos. But the big deal is TAN vs ATAN2

print "\n*** Illustrating Shadows PERL *** vDial Jan 5, 2014\n";

print "\nEnter a latitude as nn.n  ";
$latitude = 90-readline(*STDIN);      # changed to co-latitude

print "\nEnter a longitude offset (+ is west, - is east) as nn.n  ";
$longitude = readline(*STDIN);        # remove "-" as moving from h->vDial

$slat = sin($latitude*2*3.14159/360);

$hr=6;                                # change here for h->v Dial
while ($hr>-7) {                       # change here for h->v Dial

    # following does not need "use Math::Trig;" however, it is needed for ATAN etc
    # $hlaTAN = $slat ;
    # $hlaTAN = $hlaTAN * sin(15*($hr+($longitude*4/60))*2*3.14159/360) ;
    # $hlaTAN = $hlaTAN / cos(15*($hr+($longitude*4/60))*2*3.14159/360) ;

    $hlaTAN = $slat * Math::Trig::tan(15*($hr+($longitude*4/60))*2*3.14159/360);

    $hla    = atan($hlaTAN);           # needs "use Math::Trig;"
    $hla    = $hla*360/(2*3.14159);

    # http://alvinalexander.com/blog/post/perl/reference-page-perl-printf-
    # formatting-format-cheat-sheet
    print "\nhour is: " , 12-$hr , " hour line angle is: ";  # change here for h-
    >v Dial
    printf("%.2f", -$hla);             # change here for h->v Dial

    $hr = $hr - 1;                     # change here for h->v Dial
}

print "\nbye.\n";
$latitude = readline(*STDIN);          # any old input to pause screen
# end #
```

```

C:\Perl64\bin\perl.exe

*** Illustrating Shadows PERL *** vDial Jan 5, 2014
Enter a latitude as nn.n   33.5
Enter a longitude offset (<+ is west, - is east) as nn.n   7.1

hour is: 6 hour line angle is: 81.50
hour is: 7 hour line angle is: -80.55
hour is: 8 hour line angle is: -63.13
hour is: 9 hour line angle is: -46.97
hour is: 10 hour line angle is: -32.24
hour is: 11 hour line angle is: -18.71
hour is: 12 hour line angle is: -5.93
hour is: 13 hour line angle is: 6.60
hour is: 14 hour line angle is: 19.40
hour is: 15 hour line angle is: 32.99
hour is: 16 hour line angle is: 47.79
hour is: 17 hour line angle is: 64.04
hour is: 18 hour line angle is: 81.50
bye.

```

Latitude 33.5		Hour line angles for		Vertical hour line angles with long corr	
TIME OF DAY		simple dials			
am	pm	VERTICAL		am	pm
12.00	12.00	0.00		5.93	-5.93
11.50	0.50	6.27		12.25	0.33
11.00	1.00	12.60		18.71	6.60
10.50	1.50	19.06		25.35	12.94
10.00	2.00	25.71		32.24	19.40
9.50	2.50	32.61		39.43	26.07
9.00	3.00	39.82		46.97	32.99
8.50	3.50	47.38		54.87	40.22
8.00	4.00	55.30		63.13	47.79
7.50	4.50	63.59		71.72	55.74
7.00	5.00	72.19		80.55	64.04
6.50	5.50	81.03		89.52	72.65
6.00	6.00	90.00		-81.50	81.50
		These have no longitude correction		These have longitude correction	

PERL IDE

<http://open-perl-ide.sourceforge.net/>

Open_Perl_IDE_1.0.11.409.zip

<http://open-perl-ide.sourceforge.net/documentation/user-manual/usermanual.html>

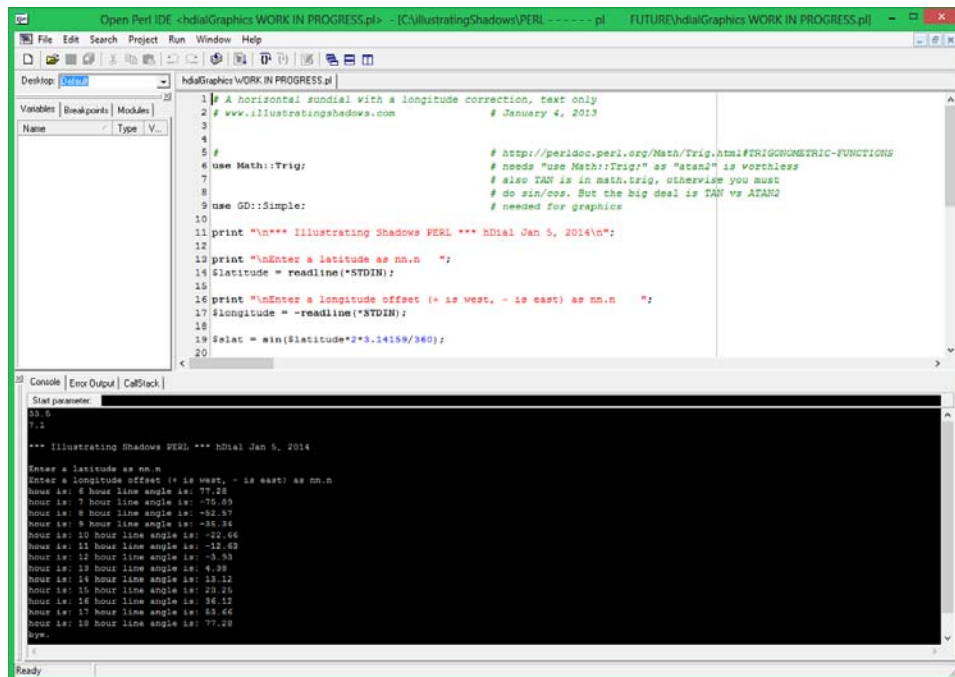
Debugging the hDial and vDial programs was done with the standard Windows Notepad or Wordpad programs. The files were saved and double clicked to run them. Syntax and other abnormal termination errors were hard to debug because the DOS screen in Windows 8 flashed up and vanished. So, an IDE would have been very helpful.

The IDE whose url is shown above installed easily. Start it with...

PerlIDE.exe

It will load and then run Perl programs (or scripts if you prefer). However, the display of printing does NOT happen until after the INPUT values are read. So, you must do:-

33.5 <enter>
7.1 <enter>
<enter>



The screenshot shows the Open Perl IDE interface. The main window displays a Perl script for calculating sundial line angles. The script includes comments about the source and the need for Math::Trig. It prompts the user for latitude and longitude, then calculates and prints the line angles for each hour of the day. The console window at the bottom shows the execution output, including the header '*** Illustrating Shadows PERL *** hDial Jan 5, 2014' and the calculated line angles for each hour, ranging from 77.28 to -22.66 degrees.

```
1 # A horizontal sundial with a longitude correction, text only
2 # www.illustratingshadows.com # January 4, 2013
3
4
5 #
6 use Math::Trig; # http://perldoc.perl.org/Math/Trig.html#TRIGONOMETRIC-FUNCTIONS
7 # needs "use Math::Trig;" as "atan2" is worthless
8 # also TAN is in math.trig, otherwise you must
9 # do sin/cos. But the big deal is TAN vs ATAN2
10 # needed for graphics
11
12 use GD::Simple;
13
14 print "\n*** Illustrating Shadows PERL *** hDial Jan 5, 2014\n";
15
16 print "\nEnter a latitude as nn.n ";
17 $latitude = readline(*STDIN);
18
19 print "\nEnter a longitude offset (+ is west, - is east) as nn.n ";
20 $longitude = -readline(*STDIN);
21
22 $slat = sin($latitude*2*3.14159/360);
```

Start parameters:
33.5
7.1
*** Illustrating Shadows PERL *** hDial Jan 5, 2014
Enter a latitude as nn.n
Enter a longitude offset (+ is west, - is east) as nn.n
hour is: 6 hour line angle is: 77.28
hour is: 7 hour line angle is: -10.89
hour is: 8 hour line angle is: -52.97
hour is: 9 hour line angle is: -35.38
hour is: 10 hour line angle is: -22.66
hour is: 11 hour line angle is: -12.63
hour is: 12 hour line angle is: -3.53
hour is: 13 hour line angle is: 4.39
hour is: 14 hour line angle is: 13.12
hour is: 15 hour line angle is: 23.26
hour is: 16 hour line angle is: 36.12
hour is: 17 hour line angle is: 53.66
hour is: 18 hour line angle is: 77.28
Now.

on blind faith, and then and only then do the results get displayed. This has the advantage that when problems occur, the "DOS window" does not flash by, instead, there is some help from the IDE.

*** END ***

Graphics and Perl ~ two choices, GD or TK

GD with GD, the primitives are simple but there is no easy way to display the output

```
http://search.cpan.org/~lds/GD-2.46/GD/Simple.pm
http://perl.about.com/od/packagesmodules/qt/perlgdsimple.htm
http://perl.about.com/gi/o.htm?zi=1/XJ&zTi=1&sdn=perl&cdn=compute&tm=36&gps=149_8_1327_699&f=00&su=p284.13.342.ip_&tt=13&bt=8&bts=8&z=8&zu=http%3A//search.cpan.org/%7Elds/GD-2.35/GD/Simple.pm
```

GD example code...

```
# http://perldoc.perl.org/Math/Trig.html#TRIGONOMETRIC-FUNCTIONS
use Math::Trig;                                # needs "use Math::Trig;" as "atan2" is worthless
                                              # also TAN is in math.trig, otherwise you must
                                              # do sin/cos. But the big deal is TAN vs ATAN2
use GD::Simple;                                # needed for graphics

# REQUIRED FOR GD GRAPHICS TO WORK:    use GD::Simple;

# create a new image
$img = GD::Simple->new(640,640);

# draw a red rectangle with blue borders
$img->bgcolor('red');
$img->fgcolor('blue');
$img->rectangle(10,10,620,620);

$img->moveTo(100,20);
$img->font('Times:italic');
$img->fontsize(18);
$img->string('Horizontal Sundial');

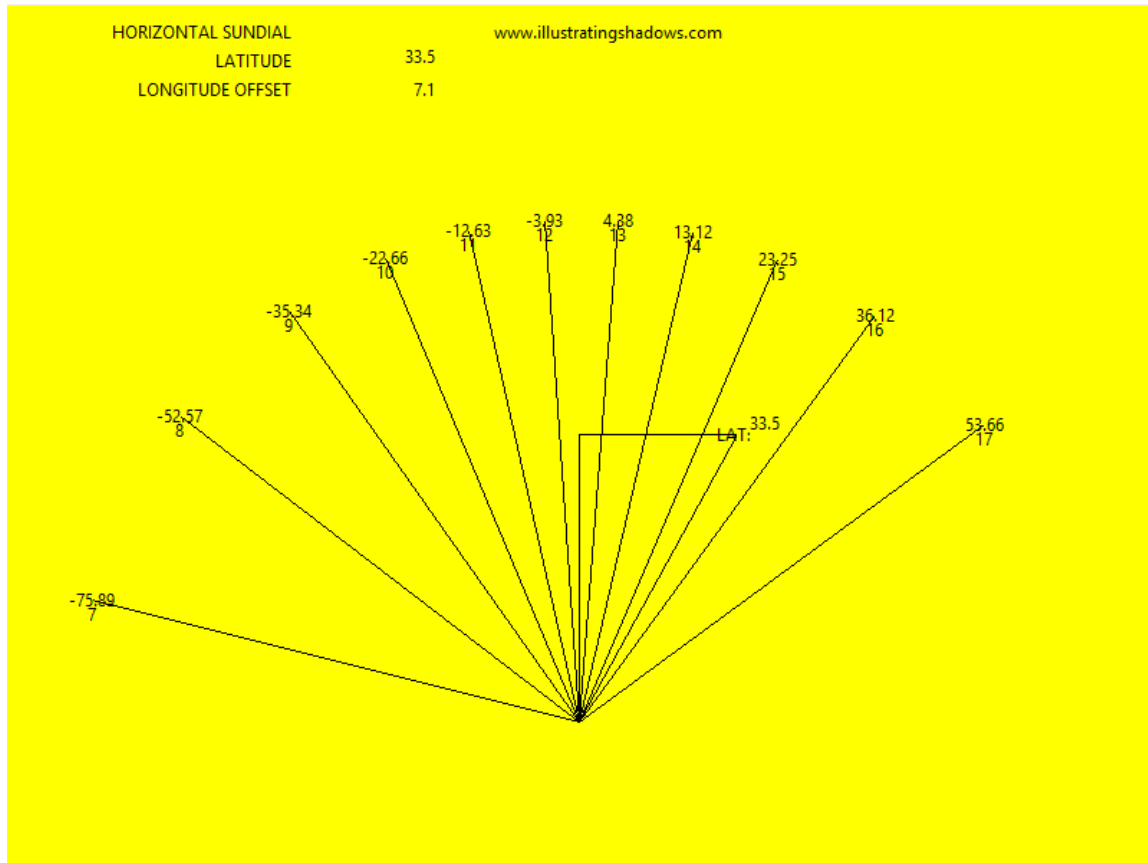
# # turtle graphics
# $img->moveTo(320,600);
# $img->penSize(3,3);
# $img->angle(0);
# $img->line(20);    # 20 pixels going to the right
# $img->turn(30);    # set turning angle to 30 degrees

$img->moveTo(320,600);
$img->penSize(3,3);
$x = 100;
$y = 100;
$img->lineTo($x,$y) ;

# convert into png data
# ##### #
# # This is where I am stuck right now . . . # #
# ##### #
print $img->png;
# write ("test.png", $img->png);

print "\nbye.\n";
# end #
```

TK example code



```
use English;
#
# http://perldoc.perl.org/Math/Trig.html#TRIGONOMETRIC-FUNCTIONS
use Math::Trig;
# needs "use Math::Trig;" as "atan2" is worthless
# also TAN is in math.trig, otherwise you must
# do sin/cos. But the big deal is TAN vs ATAN2

print "\n*** Illustrating Shadows PERL *** hDial Jan 5, 2014\n";

print "\nEnter a latitude as nn.n ";
$latitude = readline(*STDIN);

print "\nEnter a longitude offset (+ is west, - is east) as nn.n ";
$longitude = -readline(*STDIN);

$slat = sin($latitude*2*3.14159/360);

# now start the graphics stuff
use Tkx;

$mw = Tkx::widget->new(".");
# http://www.tkdocks.com/tutorial/canvas.html
```

```

$canvas = $mw->new_tk__canvas(-width => 800, -height => 600, -background =>
"yellow") ;
$canvas->g_grid(-column=>0, -row=>0);

$canvas->create_text(200,20,-text=>"HORIZONTAL SUNDIAL",-anchor=>"e");
$canvas->create_text(500,20,-text=>"www.illustratingshadows.com",-anchor=>"e");
$canvas->create_text(200,40,-text=>"LATITUDE",-anchor=>"e");
$canvas->create_text(300,45,-text=>$latitude,-anchor=>"e");
$canvas->create_text(200,60,-text=>"LONGITUDE OFFSET",-anchor=>"e");
$canvas->create_text(300,60,-text=>-$longitude,-anchor=>"e");

$hr=-5;
while ($hr<6) {                                     # NOTE: ctrl-C will break a loop in
Windows 8

    # following does not need "use Math::Trig;" however, it is needed for ATAN etc
    # $hlaTAN = $slat ;
    # $hlaTAN = $hlaTAN * sin(15*($hr+($longitude*4/60)))*2*3.14159/360) ;
    # $hlaTAN = $hlaTAN / cos(15*($hr+($longitude*4/60)))*2*3.14159/360) ;

    $hlaTAN = $slat * Math::Trig::tan(15*($hr+($longitude*4/60))*2*3.14159/360);

    $hla = atan($hlaTAN);                            # needs "use Math::Trig;" as "atan2"
is worthless
    $hla = $hla*360/(2*3.14159);

    # http://alvinalexander.com/blog/post/perl/reference-page-perl-printf-
formatting-format-cheat-sheet
    print "\nhour is: " , $hr+12 , " hour line angle is: ";
    printf("%.2f", $hla);
    $thla=sprintf("%.2f", $hla);
    #
    $xc=400;
    $yc=500;
    $radial = 350;
    $y=$radial*cos($hla*2*3.14159/360);
    $x=$radial*sin($hla*2*3.14159/360);
    $canvas->create_line($xc,$yc, $xc+$x,$yc-$y);
    $canvas->create_text($xc+$x,$yc-$y,-text=>$thla);
    $canvas->create_text($xc+$x,$yc+10-$y,-text=>(12+$hr));

    $hr = $hr + 1;
}

# draw gnomon
$canvas->create_line($xc,$yc, $xc+0,$yc-200);
$x = 200*sin($latitude*2*3.14159/360);
$canvas->create_line($xc,$yc-200, $xc+$x,$yc-200);
$canvas->create_line($xc,$yc, $xc+$x,$yc-200);
$canvas->create_text($xc+$x+10,$yc-200,-text=>"LAT:",-anchor=>"e");
$canvas->create_text($xc+$x+30,$yc-200,-text=>$latitude,-anchor=>"e");

Tkx::MainLoop();

print "\nThe End\n";

```