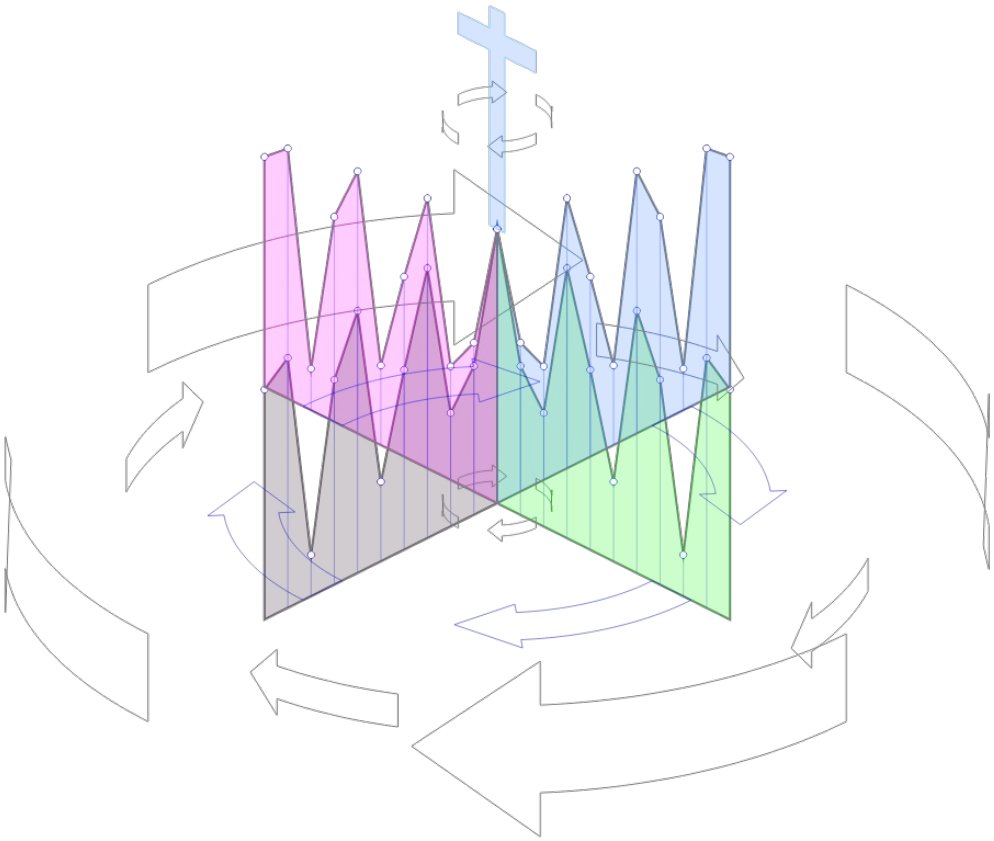
GRAFICANDO CURVAS CON JAVASCRIPT



Tutor: Juan Carlos Larios
Autor: Salvador Pina Roca
Trabajo Fin de Grado – Grado en Náutica y Transporte Marítimo – Noviembre 2016

Índice

Introducción

El hecho de realizar este trabajo responde a varios años de programación y estudio de las curvas parametricas ,curvas de Bezier y otro tipo de curvas tanto en 2d y 3d y su representación gráfica en los ordenadores utilizando un lenguaje moderno ,liviano,ejecutado por todos los navegadores y que no necesita de compilador ni de ningún entorno de programación ,que es :javascript.

Empiezo pues por definir conceptos básico,como incrustar el código de javascript en un documento HTML5 ,y resalto la importancia que tienen el conocimiento y comprensión de las formulas matemáticas, la conversión y paso en las distintas coordenadas (polares,parametricas) y decir por ultimo y no por ello menos importante que para progresar en el lenguaje de la programación la manera mas rápida y efectiva de hacerlo es progresando en la comprensión del lenguaje matemático.

Que es javascript?

JavaScript se introdujo en 1995 como una forma de agregar programas a páginas web en el navegador Netscape Navigator. La lengua ha sido adoptada desde entonces por todos los otros principales navegadores web gráficos. Ha modernizado la web que conocemos hoy día.

Características

- Javascript es un lenguaje liviano, cuya principal característica es que trabaja del lado cliente, ya que el navegador soporta la carga de procesamiento.
- Javascript puede ser insertado en páginas HTML o bien ser agregado como referencias, al igual que las hojas de estilos o CSS.

- Es un lenguaje de programación.
- No debe confundir Java con Javascript.
- Javascript por si sólo no permite la creación de aplicaciones independientes. Necesita estar inserto en un documento HTML para poder operar.
- Para programarlo sólo necesita de un editor de texto o de html que le permita editar sus documentos.

JavaScript , HTML5 y Canvas

Este apartado ofrece una breve revisión de los elementos de JavaScript y HTML5 que haremos más uso en el resto de este trabajo. No está destinado a ser un tutorial exhaustivo sobre JavaScript; En su lugar, es un resumen de lo que se necesita saber para entender los ejemplos de código de este trabajo. El otro objetivo de este apartado es cubrir aspectos relevantes del HTML5 elemento de canvas y JavaScript que establecerá el contexto para la aplicación de las fórmulas matemáticas.

Los temas cubiertos en este apartado incluyen lo siguiente:

• HTML5 y canvas: HTML5 es el último estándar de HTML, y aporta nuevas características interesantes al navegador web. La adición más importante para nuestro propósito es el elemento canvas, que permite renderizar gráficos y animaciones sin necesidad de complementos externos.

• Objetos JavaScript: Los objetos son los bloques básicos de construcción de JavaScript. "Cosas" en el real mundo se puede representar como objetos en JavaScript. Los objetos tienen propiedades. También pueden hacer cosas por medio de métodos.

• Conceptos básicos de lenguaje de JavaScript: Para completar, las construcciones básicas de JavaScript y sus sintaxis se revisan, tales como variables, tipos de datos, arrays, operadores, funciones, matemáticas, lógica, y bucles.

• Eventos e interacción del usuario: revisamos brevemente algunos conceptos básicos y sintaxis, dando ejemplos de cómo hacer que las cosas sucedan en respuesta a cambios en el programa o usuario Interacción.

• El sistema de coordenadas de canvas: Este es el equivalente de espacio en el mundo de canvas. Los objetos pueden colocarse en un elemento del canvas utilizando su contexto de representación 2D. Revisamos las diferencias Entre el sistema de coordenadas del canvas y el sistema de coordenadas cartesiano usual en matemáticas.

• La API de dibujo del canvas : La capacidad de dibujar gráficos utilizando sólo código es una herramienta poderosa especialmente cuando se combina con matemáticas y física.

HTML5, el elemento canvas y JavaScript

HTML5 es la última versión del estándar HTML, y aporta un gran numero de nuevas capacidades al navegador web.

Se necesita un conocimiento básico de HTML5 para explotar la característica más importante para fines de animación: el elemento canvas.

Esquema de un documento básico de HTML5

Para nuestros propósitos así pues, es necesario conocer un poco de HTML5. Aquí hay un ejemplo de un HTML5 básico.

```
<! Doctype html>
<html>
<head>

<Title> Un documento HTML5 básico </ title>
</ head>
<body>
<H1> Hola HTML5! </ H1>
</ body>
</ html>
```

Agregaremos algunas etiquetas más para incluir un elemento de canvas, estilo CSS y código JavaScript.

El elemento Canvas

Una de las adiciones más interesantes a la especificación HTML5 es el elemento canvas, que permite renderizar gráficos y, por lo tanto, animación, en el navegador web sin necesidad de plug-ins externos como el Flash Player.
Para agregar un elemento canvas a un documento HTML5 no podría ser más sencillo. Incluya la línea siguiente en la parte del cuerpo del documento:

```
<Canvas id = "canvas" width = "700" height = "500"> </ canvas>
```

Esto produce una instancia canvas de las dimensiones especificadas a las que se puede acceder en el Modelo de objetos de documento
(DOM) a través de su ID especificado.
Puede diseñar el canvas de la misma forma que cualquier elemento HTML normal. En el ejemplo ejemplo1.html

```
<link rel = "stylesheet" href = "style.css">
```

Si busca en el archivo style.css, encontrará que he elegido para diferentes colores de fondo para la sección de cuerpo
Y el elemento del canvas, de modo que podamos ver mejor este último contra el fondo del anterior.
No hay nada que le impida agregar más de un elemento de canvas a un solo documento HTML5.
Se puede incluso superponer diferentes instancias de canvas. Esta técnica puede resultar muy útil para ciertos fines,

Por ejemplo, para hacer una animación rápida sobre un fondo fijo. El archivo ejemplo1.html muestra un ejemplo simple de esto, con el archivo style1.css especificando el código de posicionamiento CSS requerido para los dos canvas.

```
<body>
<canvas id = "canvas" width = "700" height = "500"> </ canvas>
<script src = "bola.js"> </ script>
</ body>
</ html>
```

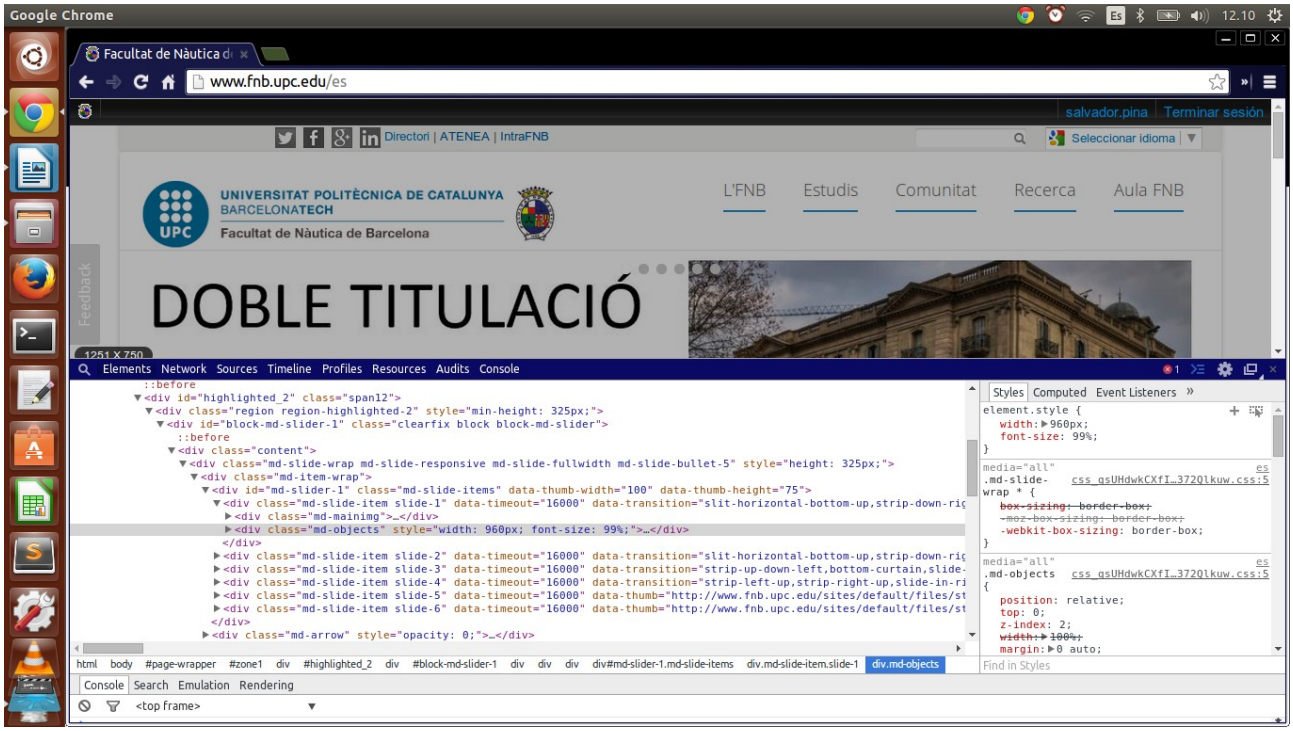
Se observa la línea de código en la parte del cuerpo del script que enlaza con el archivo bola.js, que contiene el código JavaScript. Esta línea se coloca justo antes del final de la etiqueta del cuerpo de cierre para que el DOM tenga la oportunidad de cargarse antes de ejecutar el script.

La consola de Javascript

Los navegadores modernos proporcionan una herramienta muy útil para depurar el código JavaScript, conocido como la consola. La mejor manera de aprender acerca de lo que puede hacer con la consola es experimentar con ella. Para iniciar la consola en el navegador Chrome, utilice el siguiente método abreviado de teclado: Control-Shift-J (Win / Linux) o Command-Option-J (Mac).

Puede escribir código JavaScript directamente en la línea de comandos en la consola y pulsar Intro para que se evalúe . Pruebe lo siguiente:

```
2 + 3
Console.log ( "puedo hacer JavaScript");
A = 2;
B = 3; Console.log (a * b);
```



la consola de java script del navegador Google Chrome.

Los objetos en Javascript

Si ha programado en un lenguaje de programación orientada a objetos (OOP) como C ++, Java o ActionScript 3.0,
Han sido expuestos a las clases como las construcciones fundamentales sobre las que se basan los objetos. Sin embargo, JavaScript es un lenguaje
sin clases, aunque sí tiene capacidades OOP. En JavaScript, los objetos mismos son las unidades básicas entonces, ¿qué son los objetos y por qué son útiles? Un objeto es una entidad bastante abstracta. Así que antes de definir uno, vamos a explicarlo por medio de un ejemplo. Suponga que desea crear partículas en un proyecto. Las partículas tendrán ciertas
Propiedades y podrá realizar ciertas funciones. Puede definir un objeto JavaScript general (denominado Partícula) que tiene unas propiedades y funciones. Entonces, cada vez que necesite una partícula, sólo puede crear una instancia de El objeto Partícula. Las siguientes secciones describen cómo hacer esto.

El objeto y sus propiedades

Podemos generalizar a partir del ejemplo dado para definir un objeto en JavaScript como una colección de propiedades. Una propiedad Puede a su vez ser definido como una asociación entre un nombre y un valor.

El alcance de lo que constituye el valor de un la propiedad es bastante amplio; También puede incluir funciones-vea la siguiente sección. Esto hace que los objetos sean muy versátiles.
Además de los objetos JavaScript existentes, puede crear objetos personalizados con propiedades personalizadas a voluntad.

Ejemplos de objetos predefinidos incluyen String, Array, Date y el objeto Math (que discutiremos más adelante en este capítulo).

Para crear un nuevo objeto, puede usar dos formas alternativas de sintaxis:

```
Obj = new objeto ();

Obj = {};
```

Funciones y métodos

Hemos visto cómo asignar propiedades a los objetos, pero ¿cómo podemos hacer que un objeto haga algo? Ahí es donde entran las funciones. Una función es un bloque de código que se ejecuta cuando se llama al nombre de la función. En general la sintaxis para una definición de función es la siguiente:

```
Función functionNombre () {
bloque de código
}
```

Opcionalmente, las funciones pueden llevar cualquier número de argumentos o parámetros:

```
Función functionNombre (arg1, arg2) {
bloque de código
}
```

Y pueden devolver un valor utilizando una sentencia de devolución, por ejemplo:

```
Función multiplicar (x, y) {
return x * y;
```

```
}
```

En este ejemplo, multiplique (2,3) devolverá el valor de 6.
Volviendo a los objetos, definimos un método como una propiedad de un objeto que es una función. Por lo tanto, los métodos permiten a los Objetos hacer cosas. Un método se define de la misma manera que una función, pero también se debe asignar como un propiedad de un objeto. Esto se puede hacer de varias maneras. Una sintaxis es la siguiente:

```
objetoName.methodName = functionName;
```

Por ejemplo, para asignar la función multiplicar () como una propiedad del objeto obj, podemos escribir:

```
Obj. multiplicar = multiplicar;
```

La función multiplicar es ahora un método de obj (podríamos haber usado un nombre de método diferente), y obj.Multiplicar(2,3) entonces devolverá 6. Encontraremos otras formas de asignar métodos a objetos en la siguiente sección cuando estudiamos a los constructores.

Prototipos, constructores y herencia

Un concepto importante en OOP es el de herencia, que le permite construir un nuevo objeto a partir de un objeto existente.
El nuevo objeto entonces hereda las propiedades y los métodos del objeto antiguo. En los idiomas basados en clases, la herencia se aplica a las clases, esto se conoce como herencia clásica. En JavaScript, los objetos heredan directamente de otros objetos, esto se consigue mediante un objeto interno conocido como prototipo. Por lo tanto, la herencia en JavaScript es basada en los prototipos.

El prototipo es en realidad una propiedad de cualquier función. Una función es también un objeto y por lo tanto tiene propiedades. Las propiedades atribuidas al prototipo de una función son heredadas automáticamente por objetos nuevos construidos a partir de la función objeto. Por lo tanto, un objeto de función que se pretende utilizar para construir nuevos objetos se denomina constructor.
No hay nada especial en una función constructora: cualquier función puede ser utilizada como constructor. Pero hay una convención generalizada para designar constructores por nombres de funciones que comienzan con una letra mayúscula.

He aquí un ejemplo que muestra la sintaxis en acción:

```
Función Partícula (nombre) {
This.nombre = pnombre;
This.mueve= function () {
Console.log (this.nombre + "se está moviendo");
```

Este código crea un constructor Partícula con un nombre de propiedad y un método mueve (). La palabra clave garantiza que estas propiedades son accesibles fuera del constructor. Cualquier instancia del objeto Partícula puede ser creada por la nueva palabra clave, y hereda automáticamente estas propiedades, como se muestra en este ejemplo:

```
partícula1 = new Partícula("electron");
partícula1.name; // returns "electron"
partícula1.move(); // returns "electron is moving"
```

Para agregar nuevas propiedades al objeto principal para que sean heredadas por todas las instancias del objeto, se debe asignar esas propiedades al prototipo del objeto primario. Por ejemplo, para agregar una nueva masa de propiedad y un nuevo método Stop () al objeto partícula, podemos escribir:

```
partícula.prototype.masa = 1;
partícula.prototype.stop = function () {console.log ( "He detenido");};

Estos están entonces disponibles para todas las instancias de Partícula, incluso aquellas
instanciadas anteriormente, por ejemplo:

partícula1.masa; // devuelve 1
```

Tenga en cuenta que el valor de partícula1.masa puede cambiarse posteriormente independientemente del valor predeterminado heredado de partícula.prototype.masa, por ejemplo:

```
partícula1.masa = 2;
// devuelve 2
partícula.prototype.masa; // devuelve 1;
```

Se pueden agregar otras propiedades a la instancia y, por supuesto, no se propagan al objeto principal ni a otros instancias. Por ejemplo, esta línea:

```
partícula1.spin = 0;
```

Añade una nueva propiedad llamada spin a partícula1 y le da el valor de 0. Otros casos de Partícula no tendrán esa propiedad por defecto.

Propiedades estáticas y métodos

En el ejemplo de la sección anterior, supongamos que asignamos una nueva propiedad directamente a Partícula (y no a su Prototipo), por ejemplo:

```
Tiempo de partícula = 100;
```

Esta sentencia crea una propiedad estática de partícula que es accesible sin necesidad de declarar el objeto. Por otro lado, las instancias de Partícula no heredan la propiedad estática.
Naturalmente, también se pueden definir métodos estáticos. Por ejemplo, suponga que tiene el siguiente método estático en un objeto llamado Física:

```
Función calcGravedad (masa, g) {
Retorno (masa * g);
}
Physics.calcGravedad = calcGravedad;
```

La función Physics.calcGravedad (4, 9.8) le daría entonces la fuerza de gravedad sobre un objeto de 4kg en planeta Tierra.
El objeto Math es un ejemplo de un objeto JavaScript incorporado que tiene propiedades y métodos estáticos, como Math.PI y Math.sin ().

Ejemplo: Objeto Bola

```
function Bola (radio, color) {
this.radio = radio;
this.color = color;
this.x= 0;
this.y= 0;
this.vx= 0;
this.vy= 0;
```

```
}
Bola.prototype.dibuja = function (context) {
context.fillStyle = this.color;
context.beginPath();
context.arc(this.x, this.y, this.radio, 0, 2*Math.PI, true);
context.closePath();
context.fill();
};
```

Tenga en cuenta que el objeto Bola ha recibido seis propiedades y un método. En el código de dibujo se ha colocado el método Bola.dibuja (), toma un argumento obligatorio, el contexto del canvas en el cual la pelota debe ser dibujada.

El archivo Bola-objeto.js proporciona un ejemplo sencillo de la creación de una instancia de bola desde el objeto Bola:

```
Var canvas = document.getElementById ( 'canvas');
Var context = canvas.getContext ( '2d');
Var Bola = nueva bola (50, '# 0000ff');

Bola.x = 100;
Bola.y = 100;
Bola.dibuja (contexto);
```

Haremos uso de este objeto Bola extensivamente a lo largo de estos ejemplos, haciendo varias modificaciones a lo largo de ellos.

Curvas Polares

En matemáticas,rosa polar es el nombre que recibe cualquier miembro de una familia de curvas de ecuación : $r(\theta)=k\cos\theta$ por asemejarse a una flor de pétalos.

Como casos particulares, la rosa de tres pétalos recibe también el nombre de trifolium regular y la de cuatro, el de quadrifolium. Para k=1/2 se obtiene la curva conocida como folium de Durero. Estas curvas pueden considerarse un caso especial de hipocicloide.

Estas curvas fueron estudiadas por primera vez por Guido Grandi, monje de la orden (de la Camáldulada) matemático y filósofo italiano.

Grandi presentó su estudio en Leibnitz en 1713; más tarde publicó un estudio completo en Florencia en 1728 con el título 'Flores geométrica ex rhodonearum et claeriarum curvarum '.

Luigi Guido Grandi (también conocido simplemente por el nombre que adoptó como religioso, Guido Grandi) (1 de octubre de 1671 - 4 de julio de 1742)

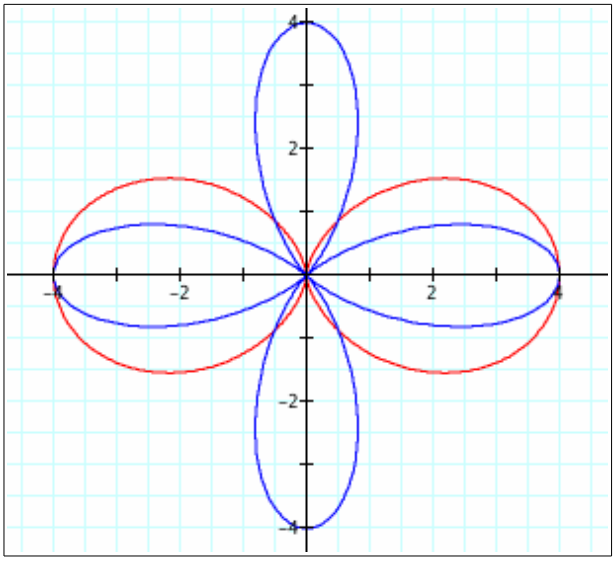
Casos generales:

Formula general de las curvas: $r = a + b\cos(k\theta)$

Caso 1: *a* = b

para a =2 y b =2.
vamos a dar diferentes valores a **k**, en este caso **k** = 2 y **k** = 4

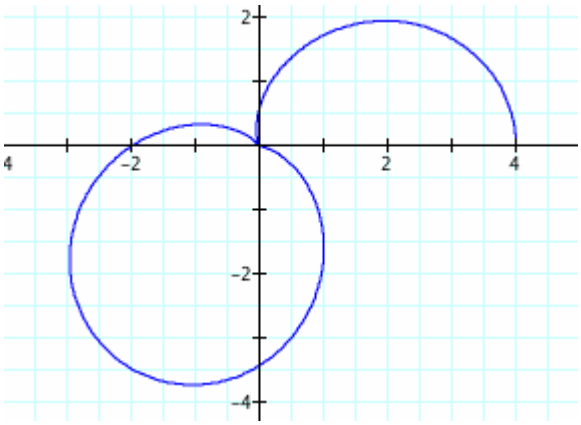
$$r = 2 + 2\cos(2\theta)$$
$$r = 2 + 2\cos(4\theta)$$



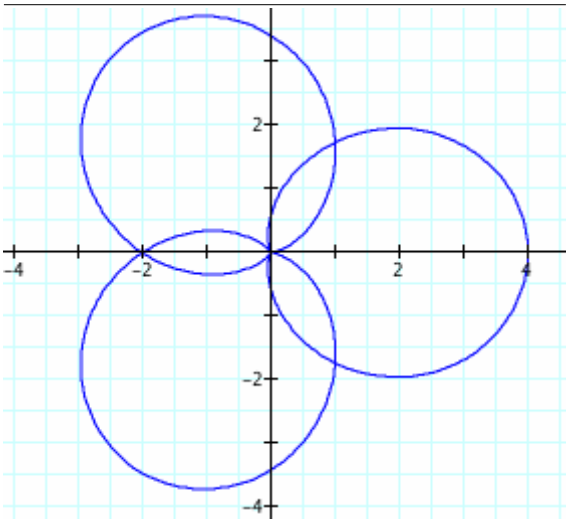
podemos comprobar que cuando k =4 hay cuatro pétalos en la flor. Cuando k=2 hay 2 pétalos.
en general cuando k es un numero entero y a = b, el gráfico obtenido es una rosa que tiene k pétalos
Si k no es un numero entero

$$r = 2 + 2\cos\left(\frac{3}{2}\theta\right)$$

La gráfica resultante,teniendo theta los valores: $[0,2\pi]$ es la siguiente:



Si cambiamos los valores de theta a $[0,4\pi]$ obtenemos el siguiente trazado:



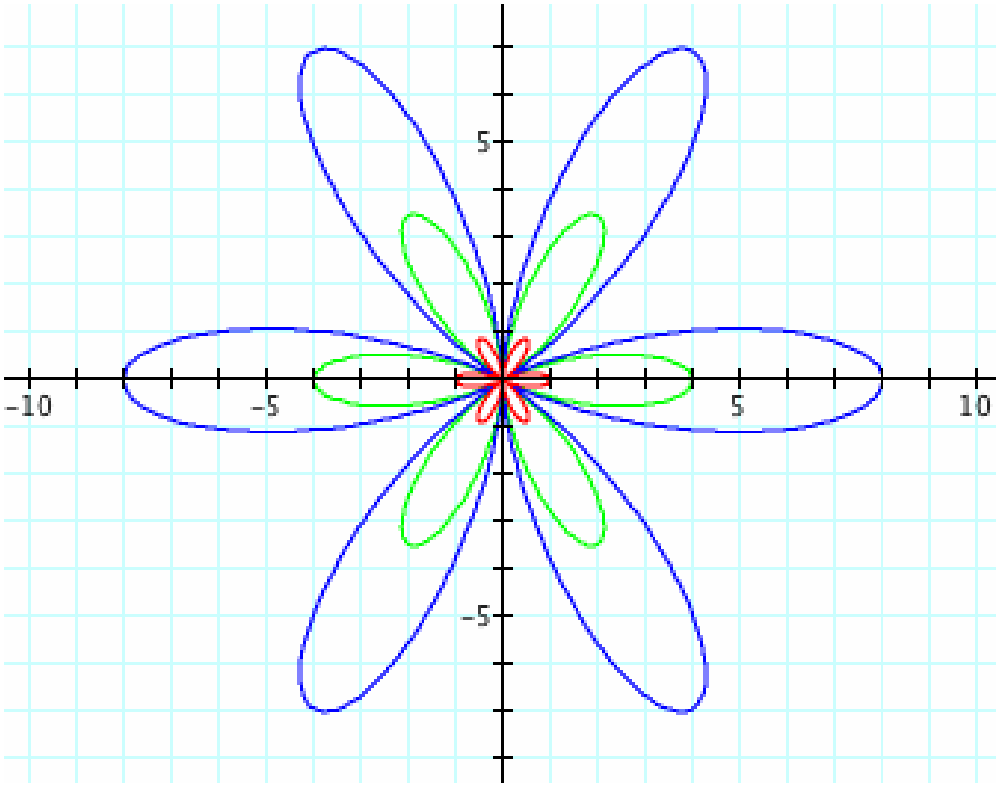
$$r = a + a\cos\left(\frac{m}{n}\theta\right)$$

Cuando k es una fracción m/n, la flor tiene m pétalos, pero los valores de theta deberán satisfacer: $[0,2n\pi]$ para poderse visualizar la gráfica por completo

$r = 2 + 2\cos(6\theta)$

$r = 4 + 4\cos(6\theta)$

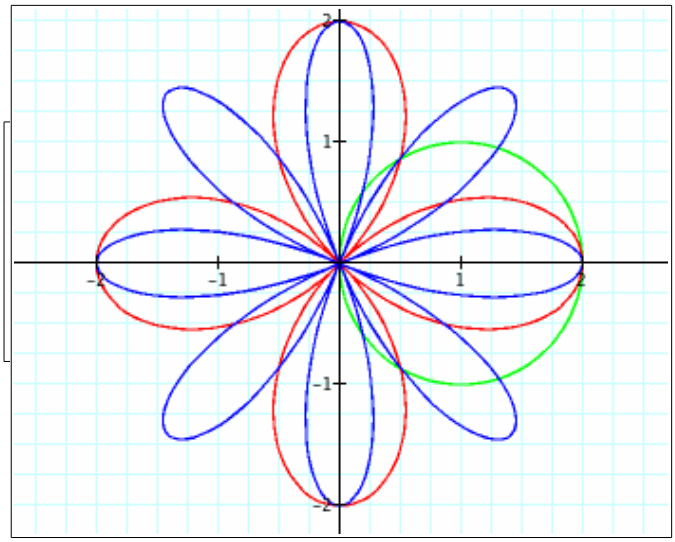
$r = 0.5 + 0.5\cos(6\theta)$

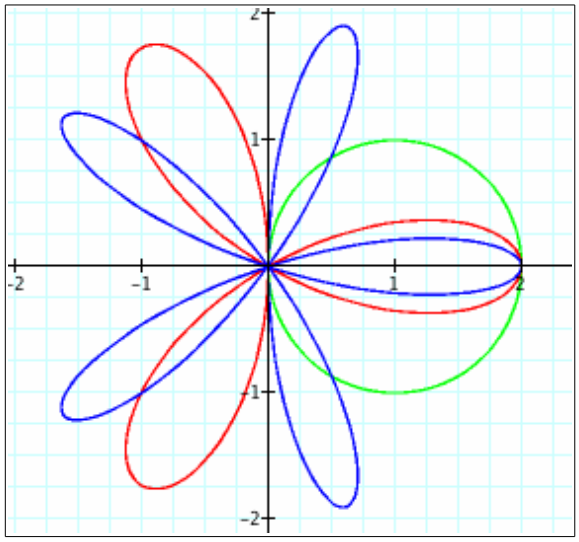


podemos comprobar que cuando a = b= 2 y k = 6, la gráfica es una
rosa de seis pétalos con longitud de 4 unidades.
Cuando a = b =4, la gráfica es una rosa de seis pétalos con longitud de 8 unidades.
Cuando a = b =0.5 , la gráfica es una rosa de seis pétalos con una longitud de 1 unidad.
Generalmente, cuando a = b , la longitud de los pétalos de la rosa es de 2 a.

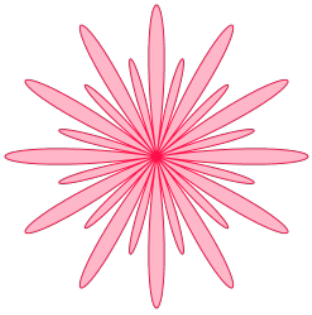
Caso 2: a < b

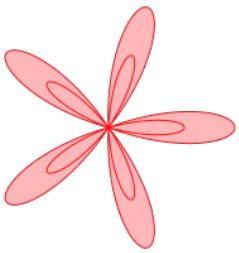
Si a=0

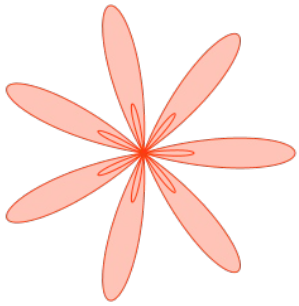




comprobamos que cuando a=0,tienen la longitud de b unidades.
Cuando k es par, hay 2k pétalos en la rosa.
Cuando k es impar, hay k pétalos en la rosa.

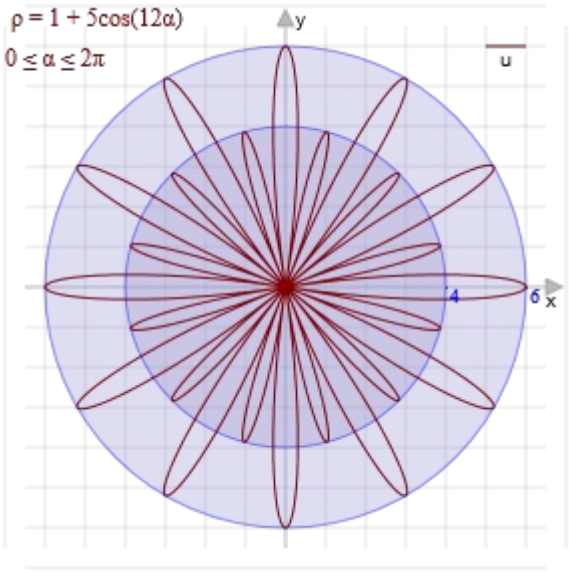


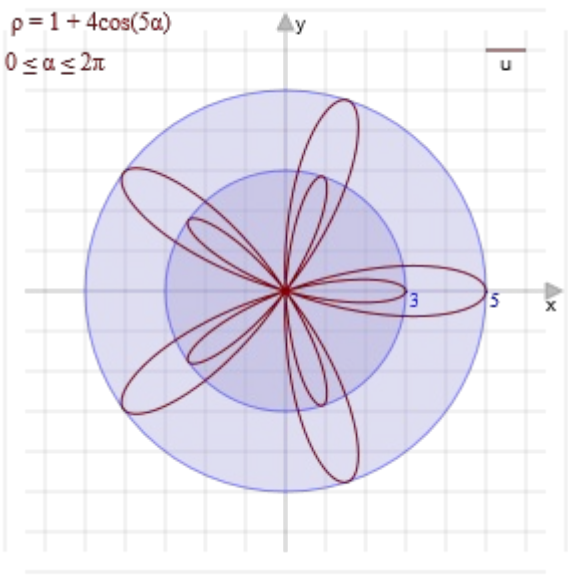




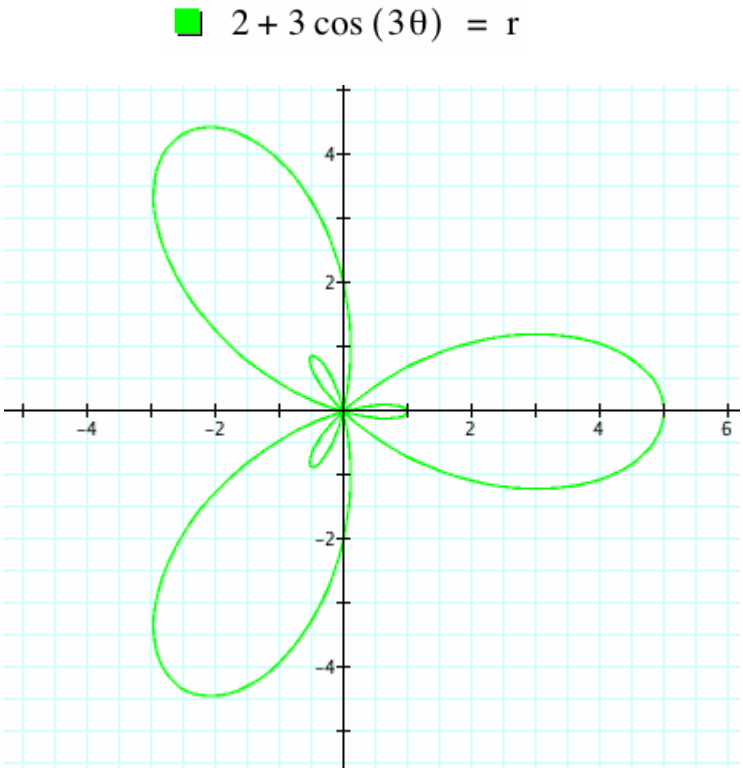
En este caso tenemos dos conjuntos de pétalos, tanto en el número de k, que están dispuestos de esta manera:

si k es impar, el menor tamaño de los pétalos, inscrito en el círculo de radio (b - a), son internos a las de mayor tamaño, inscrito en el círculo de radio (b + a);
si k es par, el menor tamaño de los pétalos, inscrito en el círculo de radio (b - a), se alternan con las de mayor tamaño, inscritos en el círculo de radio (b + a).


ρ = 1 + 5cos(12α)
0 ≤ α ≤ 2π
y
u
4
6
x


ρ = 1 + 4cos(5α)
0 ≤ α ≤ 2π
y
u
3
5
x

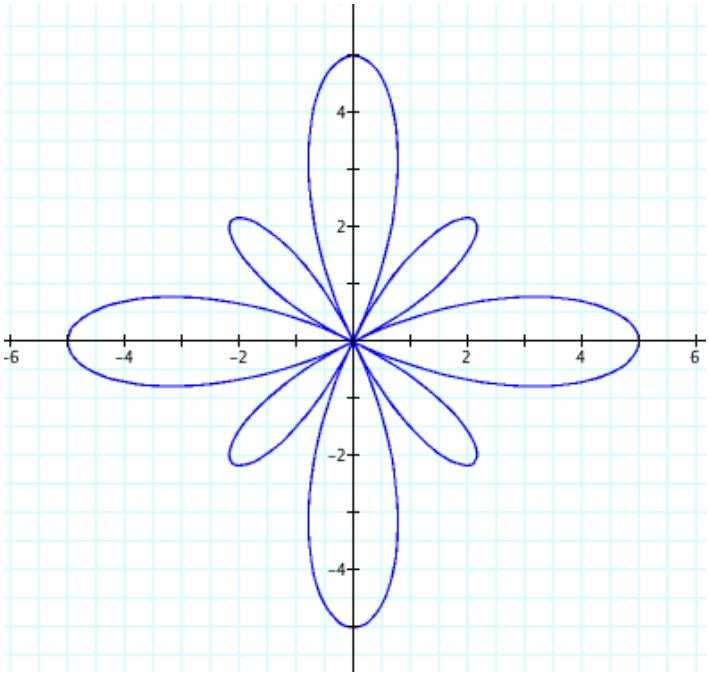
Caso a≠0


2 + 3 cos (3θ) = r
4
2
-4
-2
2
4
6
-2
-4

Comprobamos que hay dos juegos de pétalos. Hay tres pétalos más largos con la longitud de 5 unidades y 3 más cortos con la longitud de una unidad.

Otro ejemplo podría ser el siguiente:

$$1 + 4\cos(4\theta) = r$$

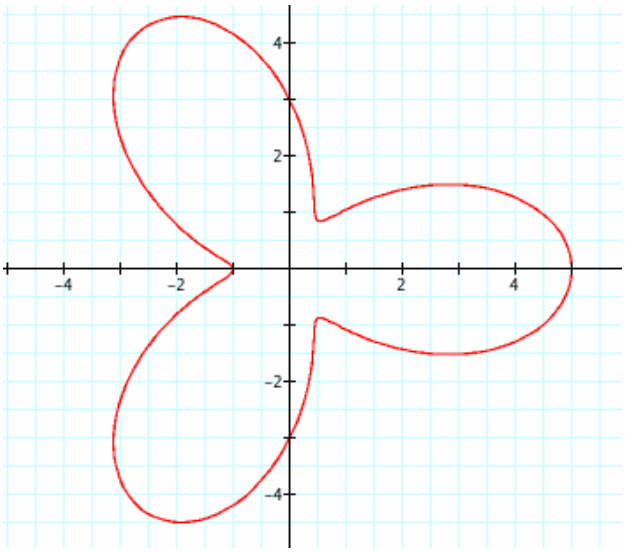


Otra vez aquí aparecen dos grupos de pétalos. Hay cuatro mas largos con la longitud de 5 unidades y 4 más pequeños entre los más grandes con una longitud de 3 unidades.

En general,cuando a < b , la flor tendrá dos grupos de pétalos.
el grupo pequeño tendrá k pétalos de longitud b – a. el grupo mayor tendrá k pétalos de longitud b + a. Cuando k es par ,pétalos pequeños se intercalarán con pétalos más grandes.
Cuando k es impar, los pétalos pequeños se encontraran encima de los grandes.

Caso 3: a > b

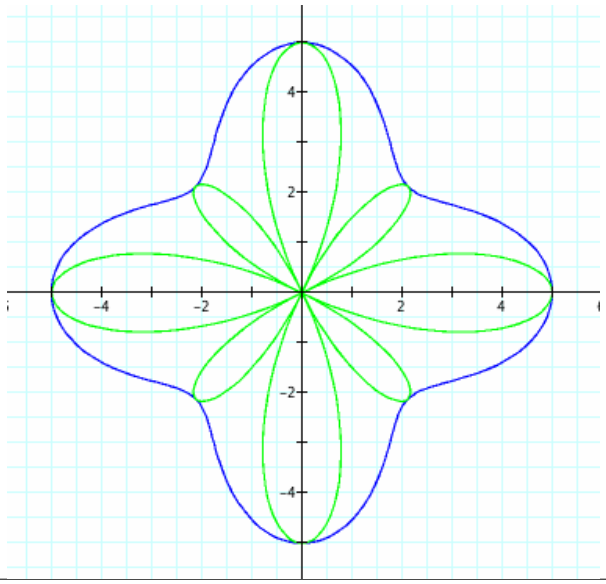
$$3 + 2\cos(3\theta) = r$$

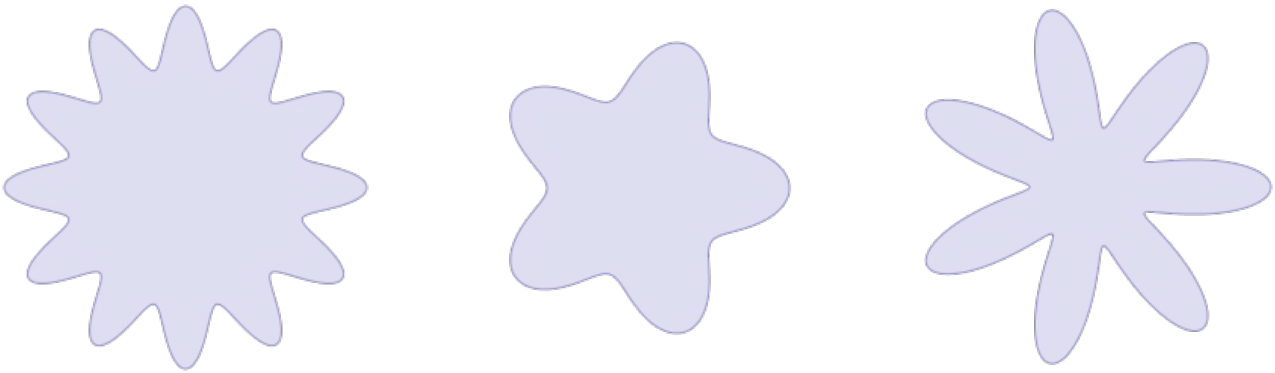


Cuando a > b , los pétalos no parten del centro
Vemos que también se cumple en esta gráfica de dos curvas polares:

$$4 + 1\cos(4\theta) = r$$

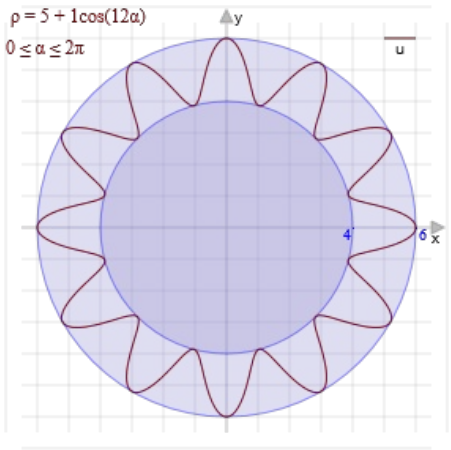
$$1 + 4\cos(4\theta) = r$$

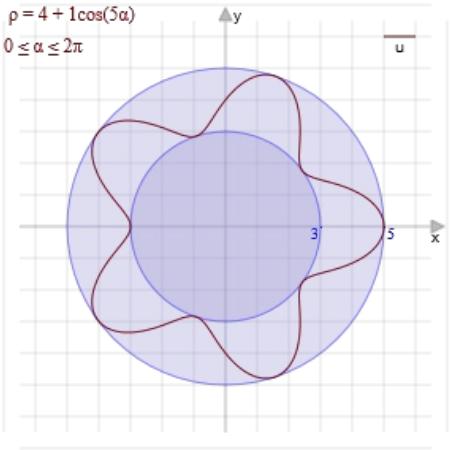


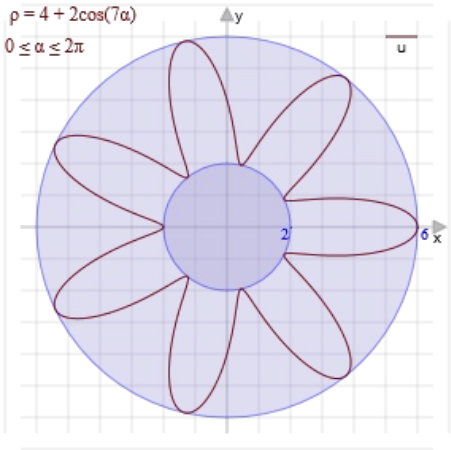


En estos ejemplos se obtiene nuevamente curvas polares que no pasan por el origen.

La curva esta comprendida entre los radios (a+b) y (a-b) , como se puede observar en los ejemplos siguientes.







En este caso tenemos dos conjuntos de pétalos, tanto en el número de k, que están dispuestos de esta manera:

si k es impar, el menor tamaño de los pétalos, inscrito en el círculo de radio (b - a), son internos a las de mayor tamaño, inscrito en el círculo de radio (b+ a);

si k es par, el menor tamaño de los pétalos, inscritos en el círculo de radio (b - a), se alternan con las

de mayor tamaño, inscritos en el círculo de radio (b +a).

Área de la curva:

Dada la ecuación polar ρ = a sin (kθ), se tiene:

$$A = \frac{1}{2}\int_0^{\pi/2k} a^2 \sin^2(k\theta)\, d\theta = \frac{a^2}{2}\int_0^{\pi/2k} \frac{1}{2}(1 - \cos(2k\theta)\, d\theta$$

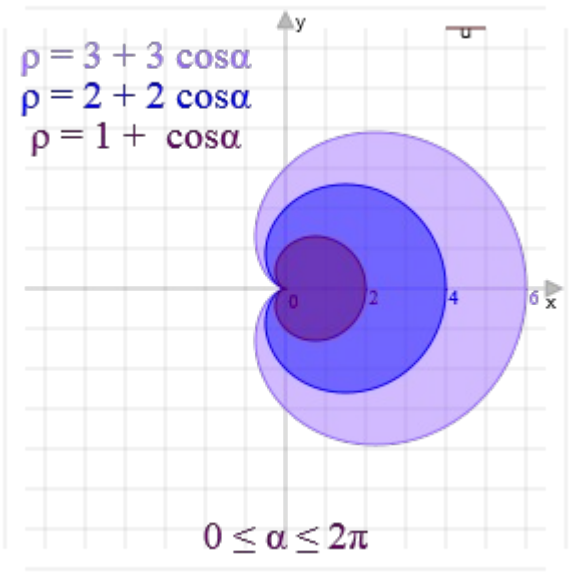
$$= \frac{a^2}{4}\left[\theta - \frac{1}{2k}\sin(2k\theta)\right]_0^{\pi/2k}$$

$$= \frac{a^2}{4}\left(\frac{\pi}{2k}\right) = \frac{\pi a^2}{8k}$$

El resultado, está tomado de Curvas-Solution, Departamento de Matemáticas, Ciencias de la Computación y Estadística Bloomsburg Universidad Bloomsburg,Pennsylvania17815
http://facstaff.bloomu.edu/skokoska/solutions.pdf

es una consecuencia de la fórmula general para derivar el área de la región limitada por la polar curva ρ = f (θ) y los rayos θ = a y θ = b; con la función continua f y positivo y 0 <b - a <2π.
La fórmula se explica en el mismo documento.
En particular, si k es impar, ya que la curva tiene k pétalos, la zona es πa2 / 4, mientras que, si k es par, ya que la curva tiene 2k pétalos, la zona es πa2 / 2.

Cardioides



Ecuación general:$\rho = a(1 \pm cos\alpha)$
a=b y k=1

El nombre, derivado de la καρδιοειδής griego (kardioeidès), en forma de corazón, se atribuyó a la curva de Johann Castillon (Castiglione del Valdarno, 11 de Enero, 1704 Artículos de - Berlín, 11 de Octubre de 1791), matemático italiano, cuyo verdadero nombre era Giovanni Salvemini, en un artículo publicado en las Philosophical Transactions de la Royal Society en 1741.

Por lo tanto, es una curva en forma de corazón

Como se puede ver desplazándose a través de las imágenes, el cardioide se puede obtener, sin una rotación con respecto al origen de los ejes, como un gráfico de varias ecuaciones polares, por ejemplo:

ρ = 1 + cosα; ρ = 1 - cosα ρ = 1 + senα; ρ = 1 - senα.

Primer método: como epicicloide

Define epicicloide, compuesto de epi (a) y cicloide, la curva generada por un punto de un círculo que rueda en la superficie exterior de otra circunferencia: es un caso particular de 'epitrocoide, desde el επι griego (epi ", en") y τροχοιδής (trocoides: en forma de rueda), curva generada por un punto de una figura que rueda sobre otra.
la cardioide se puede definir como la trayectoria de un punto fijo en la circunferencia de un círculo de radio r que está rodando sin deslizamiento alrededor de otro círculo de radio R.

Al elegir el sistema de referencia de modo que el círculo fijo tiene un centro en el punto (r, 0), se obtiene la ecuación paramétrica (1)
[X (α) = r (1 + 2cosα + cos2α)] e Y (α) = r (1 + 2senα + sen2α)]
Recordando que cos2α 2cos2α = -1 y sen2α = 2senαcosα, se obtiene:

[X (α) = 2r (1 + cosα) cosα] e [y (α) = 2r (1 + cosα) senα]

y, por último, a = 2r y α = t:
x (t) = a (1 + cost) cost
y (t) = a (1 + cost) sent
La ecuación cartesiana de la cardioide es

$$4r^2\left(x^2+y^2\right)=\left(x^2+y^2-2rx\right)^2$$

Utilizaremos las ecuaciones paramétricas de la que, en primer lugar, que :

y / x = tgα

Recordemos que:

$$cos\alpha=\pm\frac{1}{\sqrt{1+tg^2\alpha}}$$

$$cos2\alpha=\frac{1-tg^2\alpha}{1+tg^2\alpha}$$

Sustituimos estos valores en la primera de las dos ecuaciones. obtenemos

$$x=r\left(\frac{\pm2}{\sqrt{1+tg^2t}}+\frac{1-tg^2t}{1+tg^2t}+1\right)=\frac{2r\left(1\pm\sqrt{1+tg^2t}\right)}{1+tg^2t}$$

Sustituimos tgα y / x. obtenemos:

$$x=2r\frac{1\pm\sqrt{\frac{x^2+y^2}{x^2}}}{\frac{x^2+y^2}{x^2}}$$

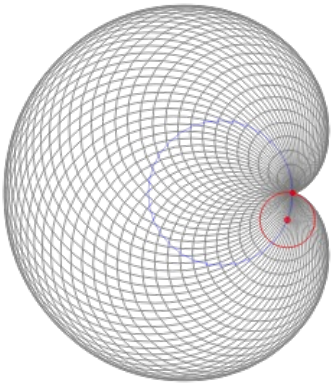
Llevamos a cabo cálculos y aislar a los radicales:

$$x^2+y^2-2rx=\pm2r\sqrt{x^2+y^2}$$

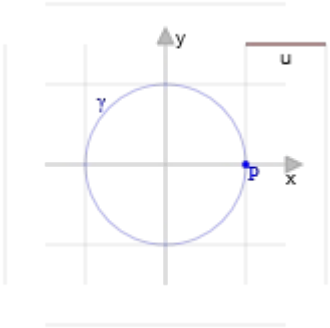
Se eleva al cuadrado para eliminar el signo doble, y aquí está la ecuación cartesiana de la cardioide:

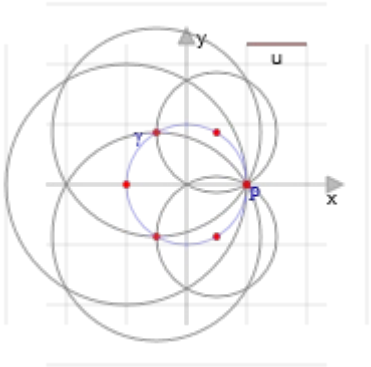
$$4r^2\left(x^2+y^2\right)=\left(x^2+y^2-2rx\right)^2$$

Segundo método: como desarrollo de una circunferencia



La cardioide puede ser construido como el desarrollo de una circunferencia, con el siguiente método.
Γ es una circunferencia fija y un punto fijo P.
La construcción de las circunferencias que tienen el centro en γ y pasar por P.
El cardioide se ha obtenido con la cúspide en el punto P y con diámetro el doble del diámetro de γ .

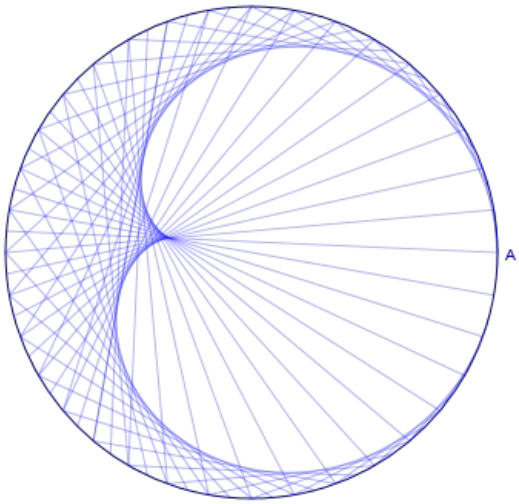




Tercer método: Regla y compás

Divida el círculo en n partes iguales y marcas de puntos asociados.
Dibujar el primer segmento (Segmento i1 j1):

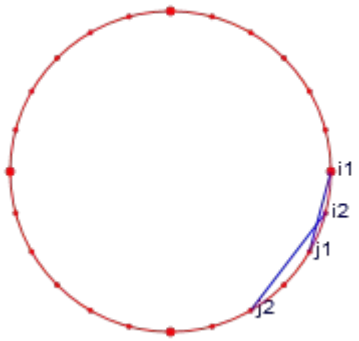
Empezar desde cualquier punto (posición i1 en la figura) que desee y unirlo con los que se hallan a una distancia de dos puntos de partida (posición J1 en la figura).

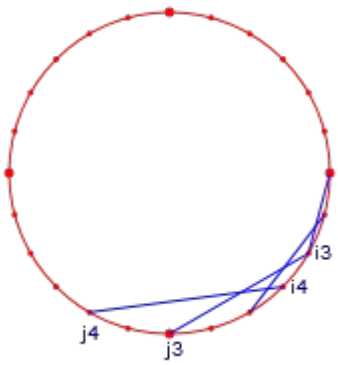


El segmento I2 J2 se construye uniendo el punto inmediatamente después de i1 con esa distancia por dos puntos J1.

El segmento de J3 i3 se construye uniendo el punto inmediatamente después de i2 con la distancia de dos puntos de J3. Continúa, el punto i - ésima ocupa la posición (i-1) + 1, mientras que el j - ésimo punto ocupa la posición (j-1) + 2

Cuando el número de puntos tiende a infinito, las cuerdas envuelven la cardioide.





Cuarto método:

1. Construir dos círculos,uno tangente al otro tal que el interior tiene un radio de 2/3 del externo.

2. Tomar un punto P en la circunferencia exterior.

3. Dibujar la linea que sale de P.

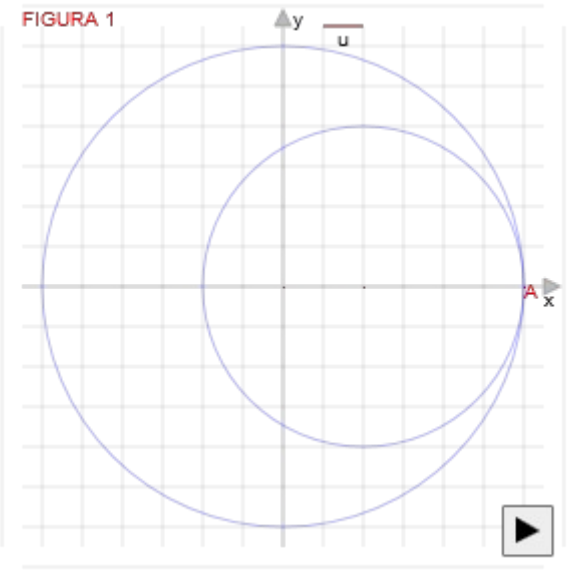
4. Dibujar la cuerda PA , donde 'A' es simétrico de A con respecto al radio OP.

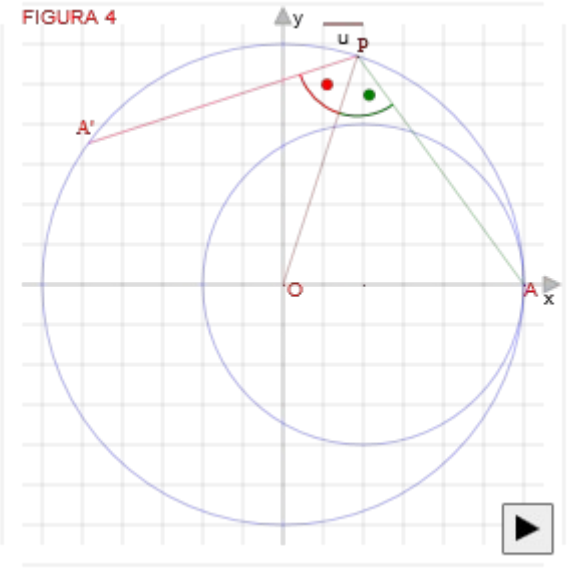
5. Dejar la cuerda AP y repetir la construcción de otros puntos de la circunferencia.

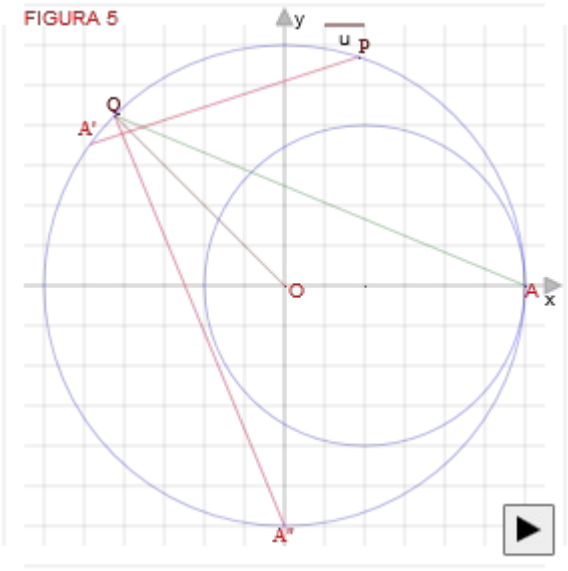
6. Repetir la construcción de un número suficiente de puntos (en el ejemplo, hemos dividido la circunferencia en 72 partes) se observa que las cuerdas muestran una cardioide que tiene la cúspide en el punto B.

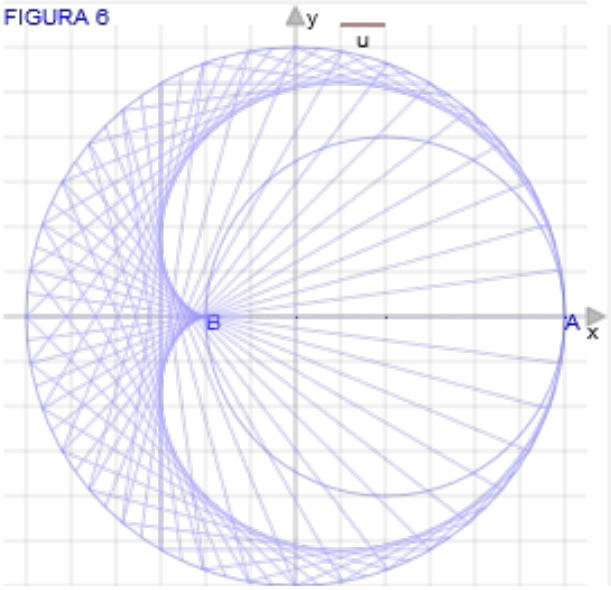
Cuando el número de puntos tiende a infinito, las cuerdas envuelven la cardioide.

Este método introduce el concepto de cardioide como reflexión cáustica









La Cáustica

En óptica, una cáustica es la envolvente de los rayos de luz reflejados o refractados por una superficie curva u objeto, o la proyección de esa envolvente de rayos en otra superficie. El término cáustica puede también referirse a la curva en la cual los rayos de luz son tangentes, determinando una frontera de la envolvente de rayos como una curva de luz concentrada. Por lo tanto en la imagen , las cáusticas pueden ser los trozos de luz o sus bordes brillantes. Estas formas normalmente tienen puntos singulares o cúspides.

Si queremos obtener la silueta de una cardioide podemos servirnos un recipiente cilíndrico que tenga las paredes reflectantes y cambiar gradualmente la inclinación con respecto a los rayos de luz incidente.

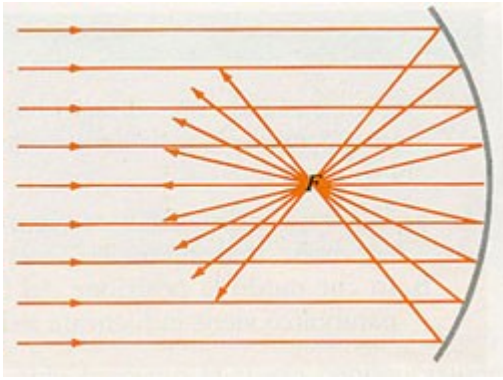
Cuando el fondo solamente es iluminado por la reflexión en las paredes internas del recipiente, que muestra una curva de la luz, cáustica de reflexión, o catacáustica, causado por la concentración de la luz en esa parte.

La imagen de la izquierda es la foto de una taza llena de agua, iluminado por los rayos del sol; la derecha, en cambio, vemos un vaso leche, en un recipiente de metal, iluminado por una lámpara.

Cuando la fuente de luz está en el exterior del cilindro, como se verá, la cáustica se convierte a una equivalente a la generada por una luz de distancia infinita.

Cardioide como cáustica de un círculo

La mayor parte de los objetos refleja una parte de la luz que incide sobre ellos y es posible verificar, cuando la superficie es lisa, la ley de la reflexión de la luz: el rayo incidente, el rayo reflejado y la normal a la superficie reflectante en el punto de incidencia se encuentran todos en el mismo plano y el ángulo de reflexión es igual al ángulo de incidencia.



En el caso de una superficie parabólica los rayos reflejados se centran en un punto, el foco, y alrededor de ella se concentra una gran cantidad de energía que puede, bajo ciertas condiciones, para calentar, o incluso prender fuego a un objeto.

Las otras curvas no tienen, en general, esta propiedad y los rayos reflejados no se concentran en un punto, si no en una curva : la cáustica

El nombre proviene de la causticus América (griego καυστός - kaustos - combustible quemado) y por lo tanto su etimología es una reminiscencia de fuego.

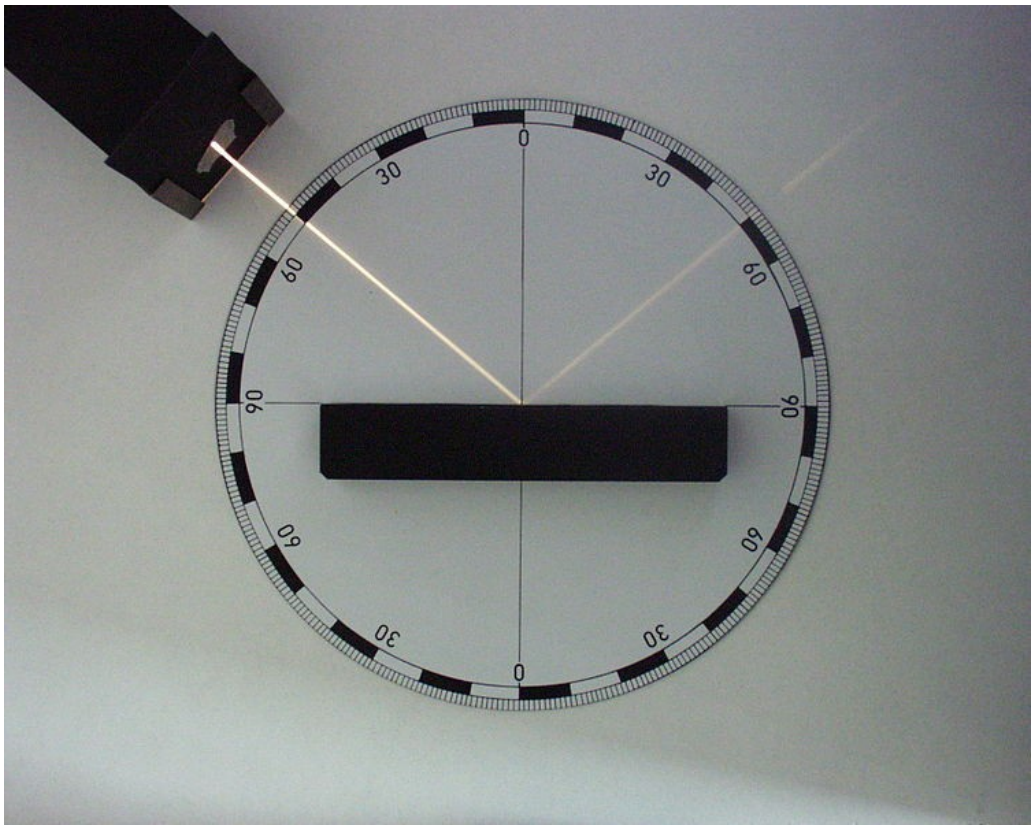
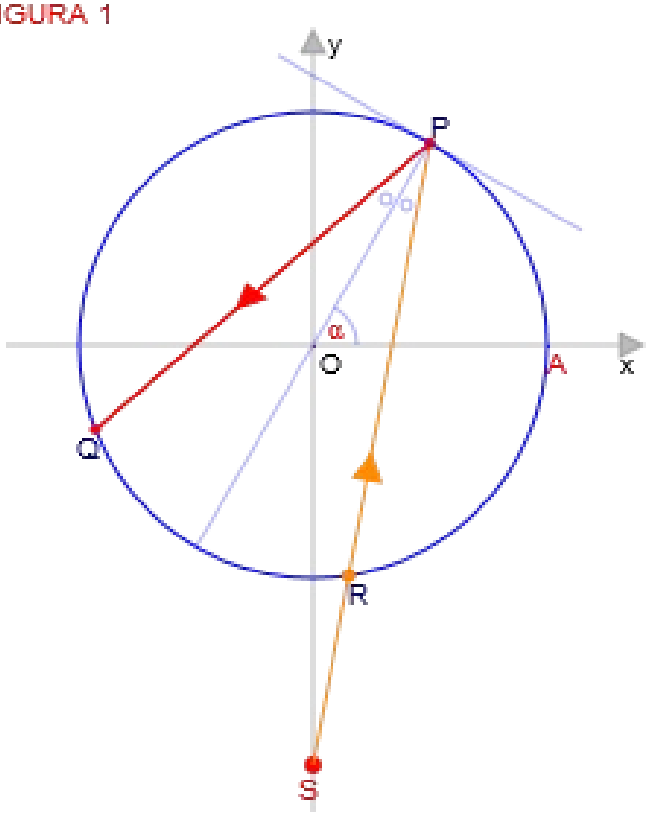


FIGURA 1



la idea para simular la caustica es la siguiente.

Tenemos una circunferencia γ de radio unitario y centrada en el origen. SP es el rayo incidente, R el punto donde el rayo incidente interseca en la circunferencia,siendo el rayo reflejado PQ simetrico dela cuerda PR respecto al radio OP.

Dado α , el angulo $\angle AOP$, para encontrar todos los puntos P bastara con variar α :

-De 0 a 2π en el caso que S pertenezca a la circunferencia,

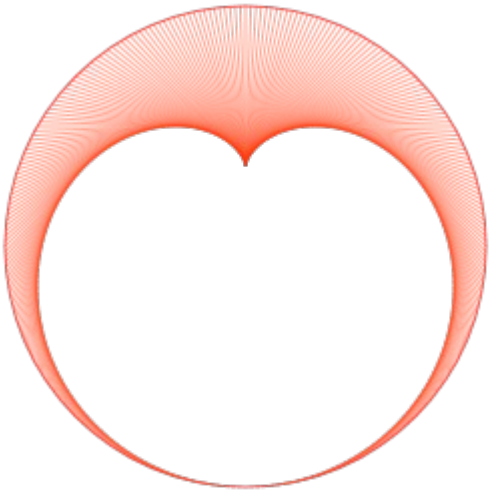
-De 0 a π en el caso en que S sea externo a la circunferencia,

$$\begin{cases} x = \frac{2cos^3\alpha}{3(1+sin\alpha)} \\ y = \frac{2sin\alpha}{3} + \frac{cos(2\alpha)}{3} \\ 0 \leq \alpha < 2\pi \end{cases}$$

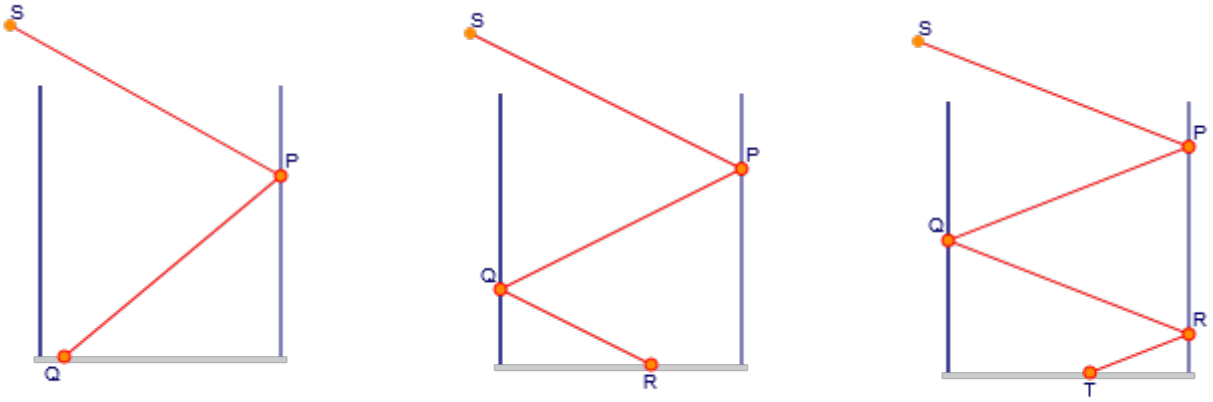
Permite encontrar, de esta manera, mediante la variación de α, el ángulo, expresado en radianes, que OP hace con el radio de la semi-eje x positivo (véase la imagen de abajo), todos los puntos que pertenecen al contorno de la cáustica.

Si rastreamos los segmentos (cosα, sinα) (x, y) se obtiene la curva que se muestra en la imagen.

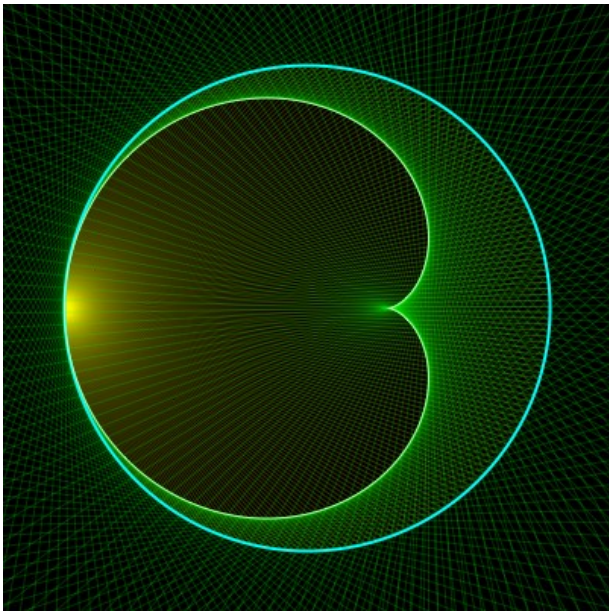
Este es el caso en el que la fuente de luz se coloca en la circunferencia.

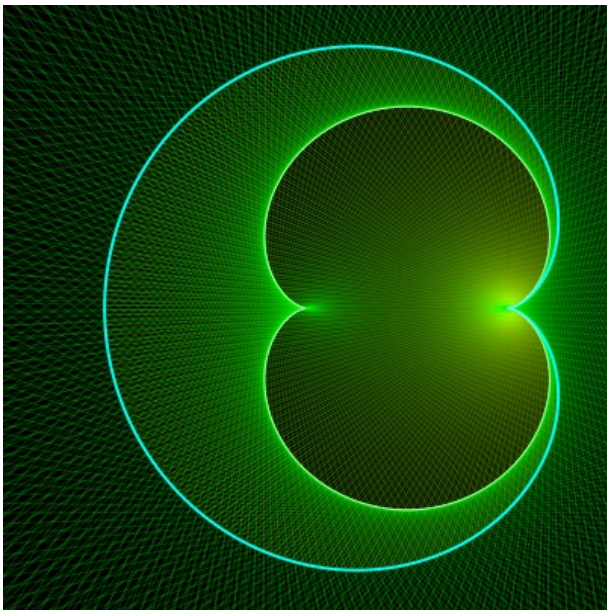


LA FORMA DE CÁUSTICA también depende del ángulo de haz incidente que forma con la superficie del recipiente contenedor



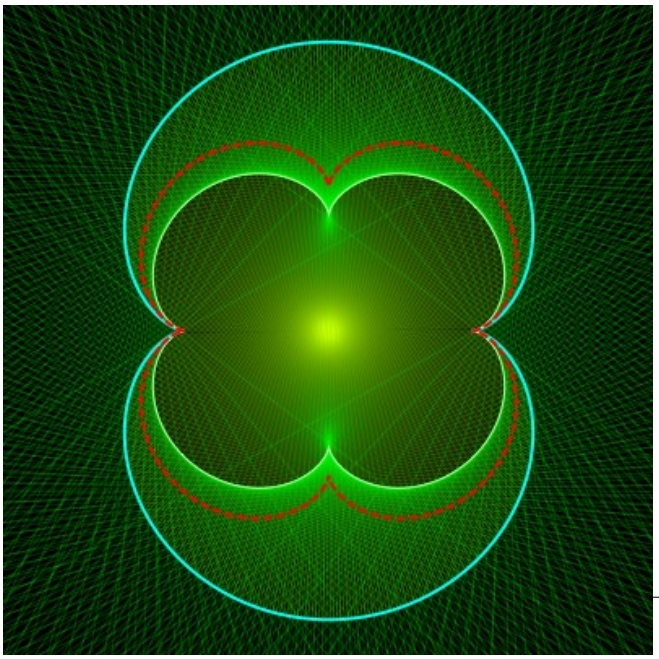
En las siguientes imágenes vemos lo que sucede cuando el rayo incidente golpea una pared lateral de un cilindro de manera que se refleje 1, 2, 3 o más veces.

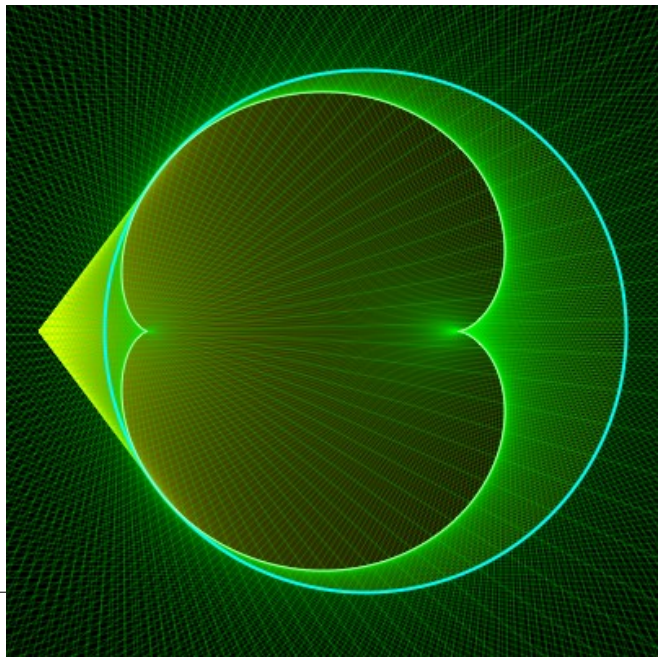




La imagen superior de la izquierda muestra la cáustica de un círculo con la fuente en su borde. Las líneas amarillas se dibujan de la fuente al espejo, y entonces los rayos reflejados se dibujan en verde (y se extienden en ambas direcciones del punto de la reflexión). La cáustica, la envolvente de los rayos reflejados, se muestra en verde claro.

La imagen superior de la derecha muestra la cáustica de un cardioide con la fuente ubicada en su cúspide, calculada usando los mismos métodos que utilizamos para el círculo. En este caso la cáustica es un nefroide.





La imagen inferior a la izquierda muestra la cáustica de un nefroide con la fuente en su centro. Esto se parece mucho a un epicicloide de 4 cúspides, pero un epicicloide real se muestra en rojo para comparación, dejando claro que las formas no son realmente las mismas.

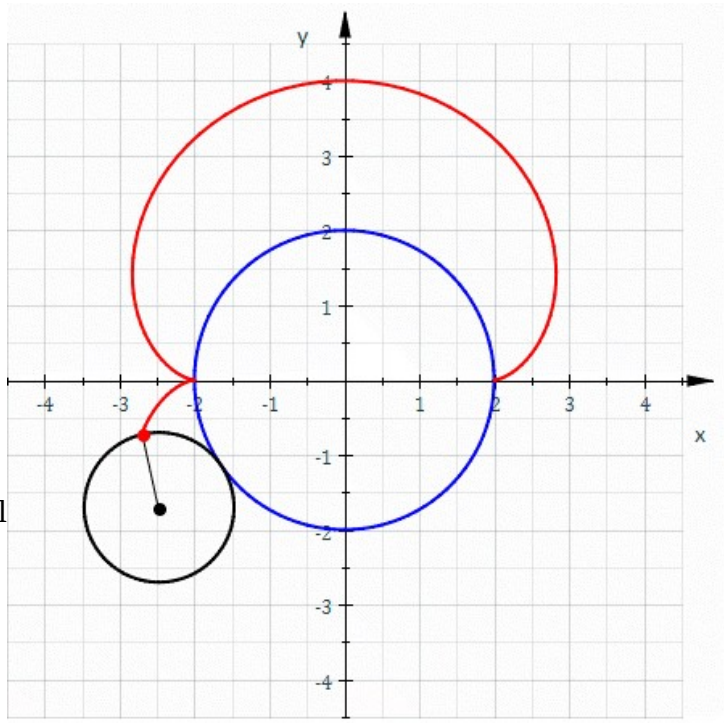
Finalmente, la imagen de abajo a la derecha muestra la cáustica de un círculo cuando la fuente está fuera del círculo. La parte de la curva que se encuentra en el lado opuesto del espejo a la fuente es una cáustica virtual: muestra la curva a lo largo de la cual los rayos reflejados por el espejo parecen originarse, aunque en realidad no habría luz presente allí.

La Cáustica por refracción

La envolvente de los rayos reflejados se llama cáustica por la reflexión, o catacáustica, para distinguirla de la cáustica obtenida por la refracción, o diacáustica, que se ve cuando los rayos procedentes de un punto penetran en un medio de diferente densidad.

Los dos fenómenos pueden darse simultáneamente.

La nefroide



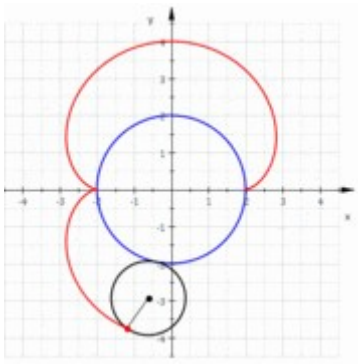
La nefroide es una curva plana cuyo nombre significa forma de riñón. Aunque el término nefroide fue usado para describir otras curvas, fue aplicado para esta por Richard Anthony Proctor (1837-1888), matemático inglés que en 1878 publicó The geometry of cycloids en Londres. Christiaan Huygens (1629-1695), en 1678, demostró que la nefroide es la catacáustica de un círculo cuando el rayo de luz está en el infinito. Publicó esto en el Traité de la lumiére en 1690. (Una explicación del porqué no se descubrió hasta que se conoció la teoría de la longitud de onda del rayo de luz; se demostró teóricamente en 1838).

La nefroide como una hipotrocoide es la línea roja

Este tipo de curvas fue estudiado además por Tschirnhausen en 1679, Jacques Bernoulli en 1692, Daniel Bernoulli en 1725 y Proctor, que como se ha mencionado le dió nombre en 1878.

Nefroide como epicicloide de dos picos (línea roja): el círculo azul de radio 3 a es tangente interior al círculo negro de radio 2a. Según el círculo mayor rueda alrededor del pequeño sin deslizar, de forma que los dos arcos rojos son siempre de la misma longitud, el punto verde traza una nefroide.

Curvas derivadas de la nefroide



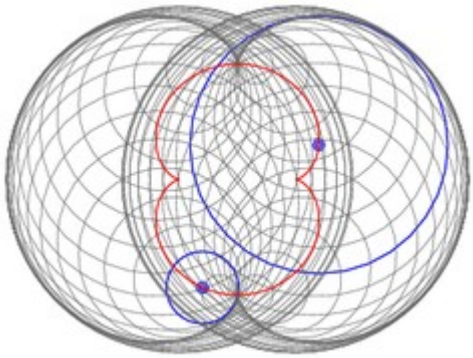
Evoluta: la evoluta de la nefroide es

otra nefroide de la mitad de tamaño y girada 90 grados. La nefroide original es así vista como una envolvente de sus círculososculadores (gris). 2 círculos osculadores pueden verse en azul. Sus centros aparecen sobre la evoluta (nefroide roja pequeña:).

involuta: como la involuta de una nefroide es otra nefroide, se tiene que la involuta de la nefroide también es otra nefroide. La nefroide original (envolvente de círculos grises) en la imagen previa, es la involuta de la nefroide roja más pequeña.

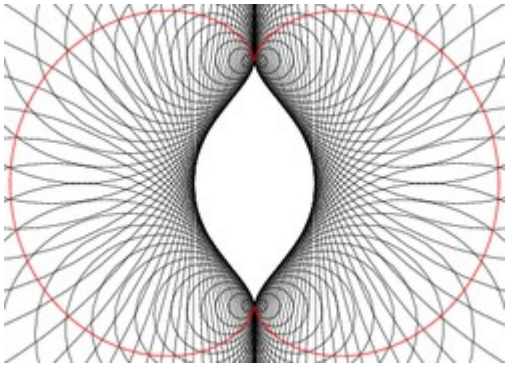
inversa: La inversa con respecto al origen de la nefroide (en rojo) es la curva mostrada como la envolvente de círculos (en negro). Estos círculos son los inversos de aquellos que servían en una imagen anterior para describir la nefroide como una envolvente de círculos. Al invertir respecto a un círculo de radio a, una ecuación cartesiana para la inversa es

Derivadas de la nefroide:



Evoluta de la nefroide (hay 60 círculos correspondientes al parámetro t tomando como valores los múltiplos de 6.)

Inversa de la nefroide



Curvas Ciclónicas

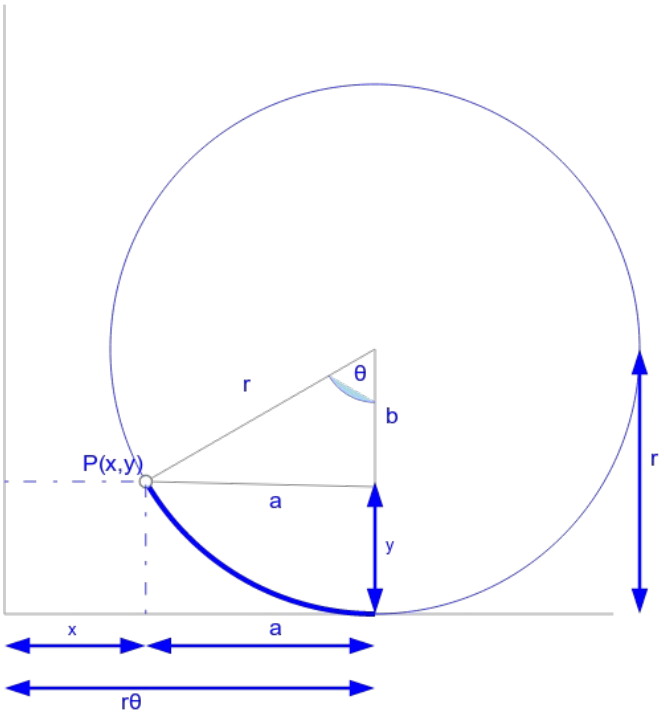
Curvas Cicloides	-Ciclónicas	natural Trocoides alargada acortada
	-Epicicloide	natural Epitrocoides alargada acortada
	-Hipocicloide	natural Hipotrocoides alargada acortada

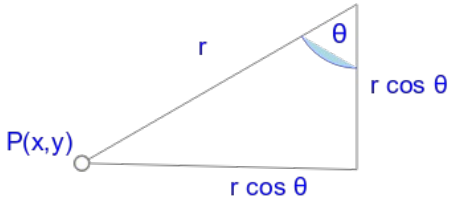
La Generatriz y la Directriz

La generatriz es una línea que a causa de su movimiento conforma una figura geométrica, que a su vez depende de la directriz. La generatriz puede ser una linea recta o curva simplemente un circulo. Si la generatriz es una línea recta que gira respecto de otra recta directriz, llamada eje de rotación, conformara una superficie cónica, cilíndrica, etc. Si la generatriz es una curva, genera esferas, elipsoides, etc. Si se desplaza sobre una o mas directrices, genera una superficie reglada. La generatriz puede ser una linea curva, por ejemplo, una circunferencia que rueda sobre otra circunferencia directriz, tangencialmente. Un punto vinculado a ella describe una trayectoria curva que se denomina ruleta cicloidal.

Cicloides

Una cicloide es el lugar geométrico originado por un punto de una circunferencia (generatriz) al rodar sobre una linea recta (directriz), sin deslizarse.





El espacio que recorre la rueda al girar queda determinado por ***r θ*** luego la coordenada ***x*** que describe el punto que gira en ella sin resbalar viene determinada por la distancia ***avance − sen θ r***

avance = rθ

a = r sen θ
x = avance - a;

por tanto la cordenada y será:

b = r cos θ
y = r - b;

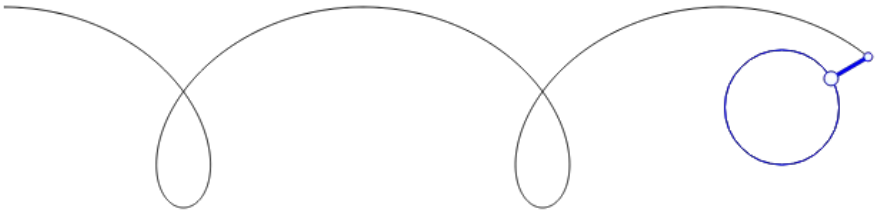
$x = r\,\theta - r\,\mathrm{sen}\,\theta \qquad y = r - r\cos\theta$
$x = r\,(\theta - \mathrm{sen}\,\theta) \qquad y = r\,(1 - \cos\theta)$
que son las ecuaciones paramétricas que describen la trayectoria de la cicloide.

Código General de las Cicloides

```
var c,ctx;
var inc=0,add=10,rad=Math.PI/180,n=0,cx,cy;
var ancho,alto;
var phi = 0;

function init(){

c = document.getElementById("c");
ctx = c.getContext("2d");
n=3;
ancho=c.width;
alto=c.height;
cx =100;
cy =alto;
dibuja();
}

function dibuja(){

inc=inc+add;
if(inc>360*n||inc<0){add=-add};
phi=inc*rad;
cicloide(ctx,{x:cx,y:cy},40,90*rad);
setTimeout(init,69);
}



function point(ctx,objeto,radio,color1,color2){
ctx.save();
ctx.strokeStyle=color1;
ctx.fillStyle=color2;
ctx.beginPath();
//ctx.moveTo(objeto.x,objeto.y);
 ctx.arc(objeto.x,objeto.y,radio, 0, 2 * Math.PI);
if(color2) {ctx.fill()} else{ctx.stroke()};
ctx.stroke();
ctx.restore();
}
```

script.4 cicloide natural

script.5 cicloide alargada



script.6 cicloide acortada

Código trocoide acortada:

```
function trocoideAcortada(ctx,centro,a,b,desfase){

var phi=inc*rad;
ctx.clearRect(0, 0,ancho,alto);

ctx.beginPath();

for (var t = 0; t <= phi; t += .01) {
var x = centro.x + (a*t)+(b*Math.sin(t+desfase)) ;//trocoide
var y = centro.y- a -((b*Math.cos(t+desfase)));
var f=a;
var x2 =centro.x + (a*t)+(f*Math.sin(t+desfase)) ;//trocoide
 var y2 =centro.y- a -(f*Math.cos(t+desfase));

ctx.lineTo(x, y);
}
ctx.stroke();
line(ctx,{x:x,y:y},{x:x2,y:y2},3,'blue');
point(ctx,{x:x,y:y},3,'blue','white');

point(ctx,{x:centro.x+(a*phi),y:centro.y-a},a,'blue');
point(ctx,{x:x2,y:y2},5,'blue','white');
}
```

Epicicloides

Las curvas epicicloides han sido estudiadas por muchos matemáticos alrededor del siglo 17: Durero (1515), Desargues (1640), Huygens (1679), Leibniz, Newton (1686), de L'Hôpital (1690), Jakob Bernoulli (1690), La Hire (1694), Johann Bernoulli (1695), Danilel Bernoulli (1725) y Euler (1745, 1781).

Apolonio de Perga (200 aC), tuvo la idea de describir los movimientos celestes como combinaciones de movimientos circulares. Fue Hiparco de Nicea (alrededor de 150 aC), el más grande astrónomo de la antigüedad griega, que trabajó a cabo esta teoría en detalle. Los resultados se volvieron famosos por los libros de Tolomeo (alrededor de 150 dC). La tierra se piensa como un obstáculo en (o cerca) un centro celeste, en torno al cual giran los demás cuerpos celestes. La combinación de la rotación de la tierra y la rotación del planeta alrededor de ella hace un epicicloide. Esta teoría geocéntrica debe ser la teoría aceptada por casi 2000 años. La teoría heliocéntrica (como se construye por Copér nico), también fue discutido por el griego, pero se negó por razones emocionales.

La curva se forma por el lugar de un punto, que se adjunta a un círculo, que rueda en 1) el exterior de otro círculo 2). En la ecuación de la curva de la primera parte indica la posición relativa entre los dos círculos, la segunda parte indica la rotación del círculo rodante.

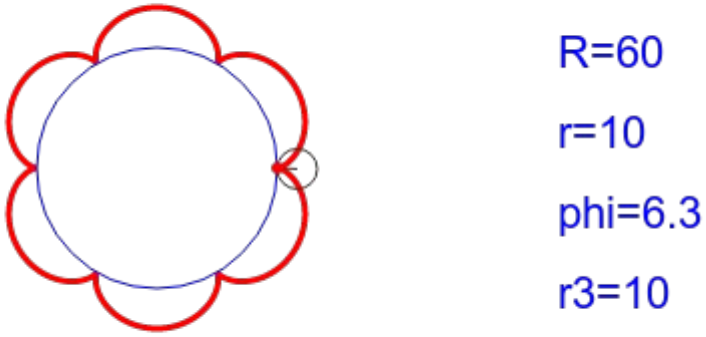
El valor de la constante b determina el punto de partida en relación con el círculo: (Ordinaria) epicicloide El punto de partida se sitúa en el círculo rodante (b = 1). Cuando el punto de partida no está en el círculo, la curva se denomina epitrocoide: epicicloide alargada El punto de partida se encuentra fuera del círculo rodante (b> 1). acortada epicicloide El punto de partida se encuentra en el interior del círculo rodante (b <1).

El isóptica del epicicloide ordinaria es un epitrocoide.
La curva tiene una forma cerrada cuando la relación del círculo de rodadura y el otro círculo (a) es igual a un número racional. Cuando se da esta relación su forma más simple, el numerador es el número de revoluciones alrededor del círculo de reposo, antes de la curva se cierra. El denominador es el número de rotaciones del círculo rodante antes de que esto suceda. En este caso la curva racional es algebraica, de otro modo trascendental. A epicicloide con parámetros a y b es el mismo que un hipocicloide con los parámetros a + 1, 1 / b.

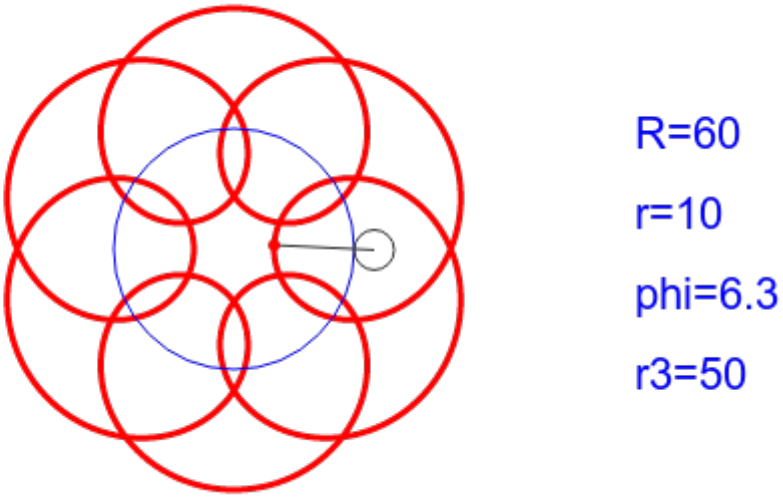
$$x=(R+r)\cos\alpha-d\cos\left(\frac{(R+r)}{r}\right)\alpha$$

$$y=(R+r)\,sen\,\alpha-d\,sen\left(\frac{(R+r)}{r}\right)\alpha$$

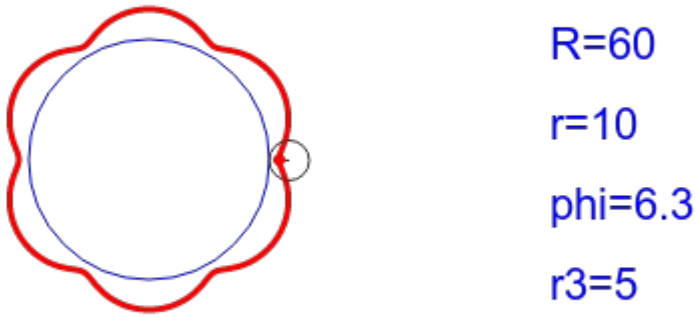
script.5 Epicicloide natural



script.6 Epicicloide alargada

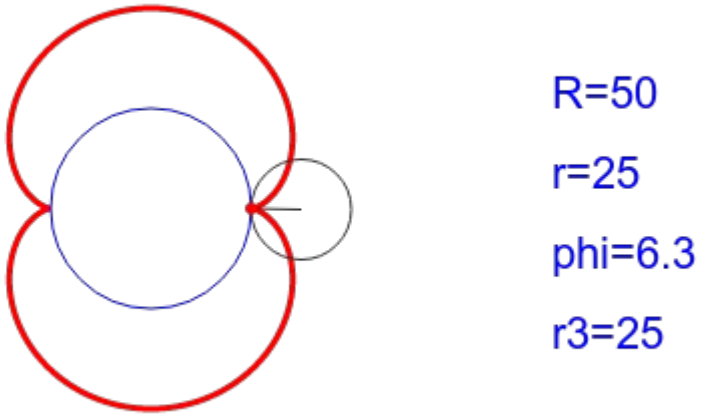


script.7 Epicicloide acortada

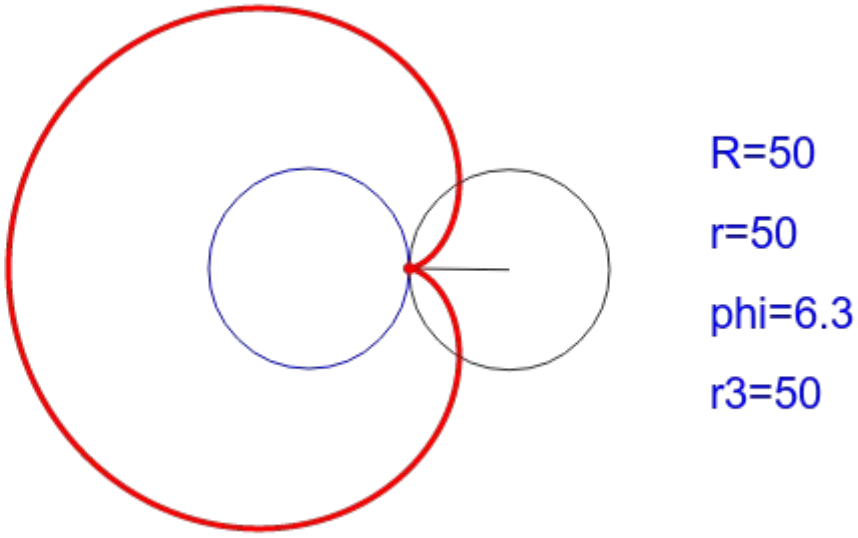


Epicicloides Notables

Script.8 Epicicloide nefroide

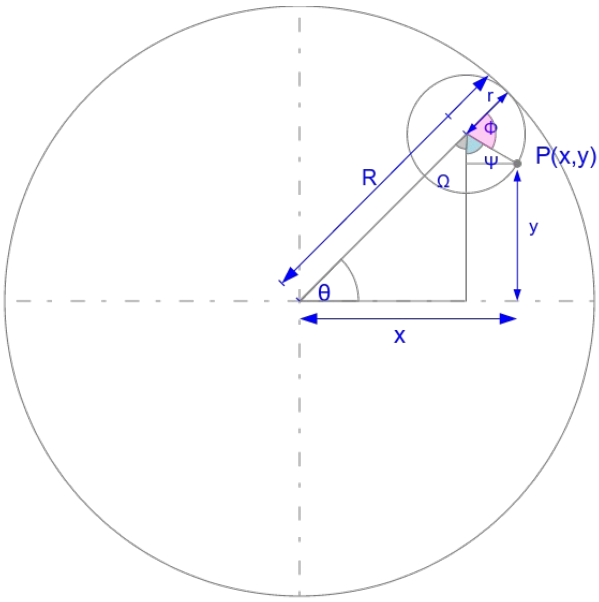


script.9 Epicicloide Cardioide



Hipocicloides

Script.4 Formulas Hipocicloide



De las razones trigonométricas de la suma de angulos obtenemos:

sen (90o+ x) = sen 90 cos x + cos 90 senx = sen 90 cosx = cos x
cos (90o+ x) = cos 90 cos x − sen 90 senx = -sen x

Vamos a obtener las coordenadas del punto P(*x,y*) que es el punto que gira en la pequeña rueda que se desplaza sin **resbalar** y que con su movimiento describe la hipocicloide.
Siendo **R** = radio mayor ; **r** = radio menor ; la hipotenusa del triangulo mayor valdra **R - r** por consiguiente, la coordenada x de P valdrá el cateto del triangulo mayor ***(R-r) cos θ*** menos el cateto del triangulo menor que vale ***rsen θ***

De los ángulos interiores del triangulo de la circunferencia menor tenemos:

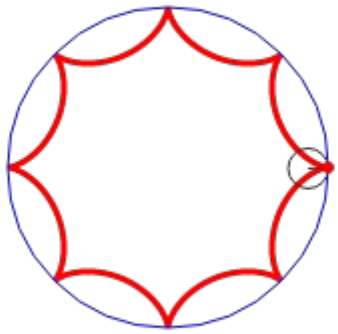
θ + φ + α =180
θ + φ +(90− α) =180
θ = φ +(90+ φ − α)

x = (R-r)cos θ+ rsen φ
x = (R-r)cos θ+ rsen [90+ (φ − θ)]

$$x=(R-r)\cos\alpha+r\cos\left(\frac{(R-r)}{r}\right)\alpha$$

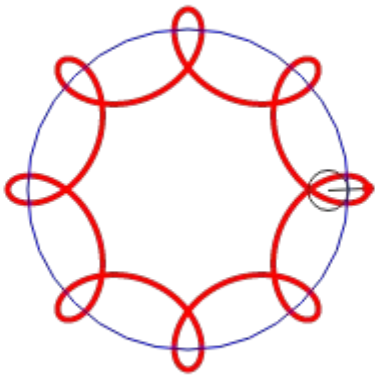
$$y=(R-r)\,sen\,\alpha-r\,sen\left(\frac{(R-r)}{r}\right)\alpha$$

script.10 Hipocicloide natural



codigo:

```
function formula(c,R,r,phi,d){
x = c.x + (R-r)*Math.cos(phi)+d*Math.cos((R-r)/r*phi);
y = c.y + (R-r)*Math.sin(phi)-d*Math.sin((R-r)/r*phi);
return{x:x,y:y};
}
```



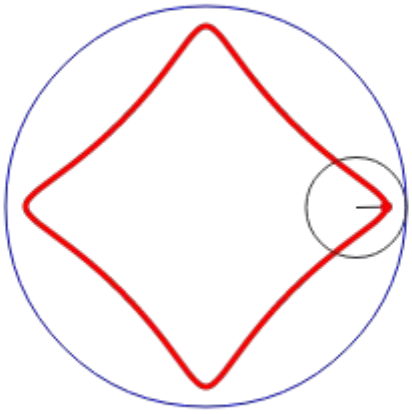
R=80

r=10

phi=6.3

r3=20

script.12 Hipocicloide Acortada



R=100

r=25

phi=6.3

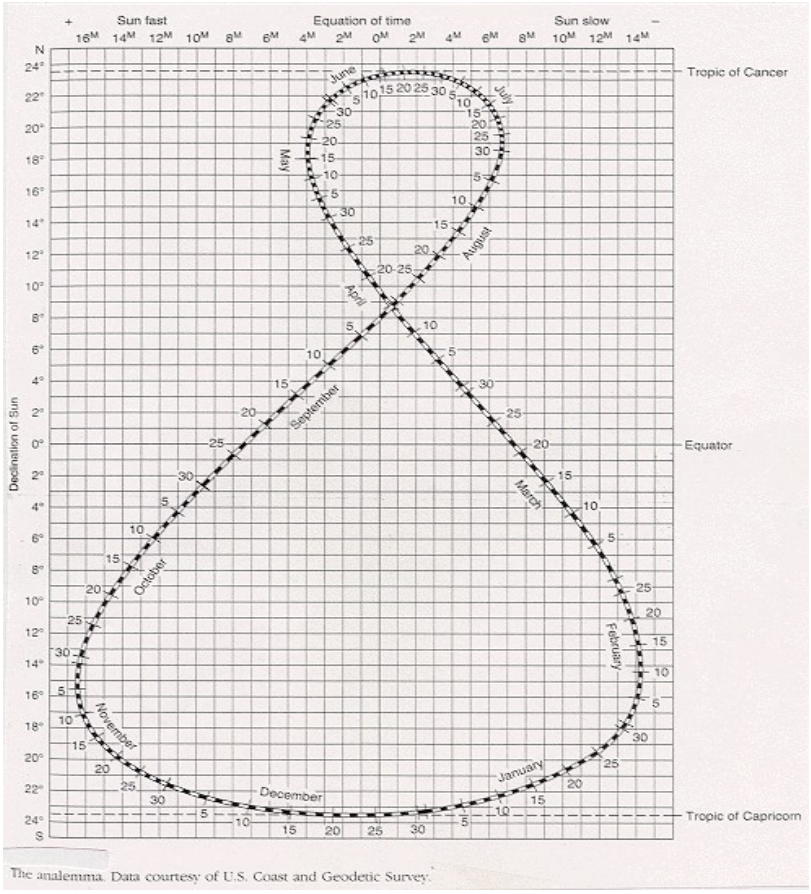
r3=15

Otras curvas:

La lemniscata

Una lemniscata es el lugar geométrico de los puntos del plano tales que el producto de sus distancias a dos puntos fijos llamados focos es constante.
Fue en el año 1694 cuando Jakob Bernoulli describió esta curva y la llamo lemniscus, que en Latín significa "cinta colgante"
Es muy interesante su construcción que esta relacionada con los óvalos de Cassini .

De forma aproximada, también está¡ presente en la naturaleza. El analema, que es la curva descrita por la posición del Sol observada todos los días del año a la misma hora y desde la misma posición, se asemeja a una lemniscata.
Si graficamos la Declinación del Sol versus la Ecuación del Tiempo para cada dí del año, obtenemos el Analema; una figura que se encuentra en muchos relojes de Sol, mapas y globos geográficos.



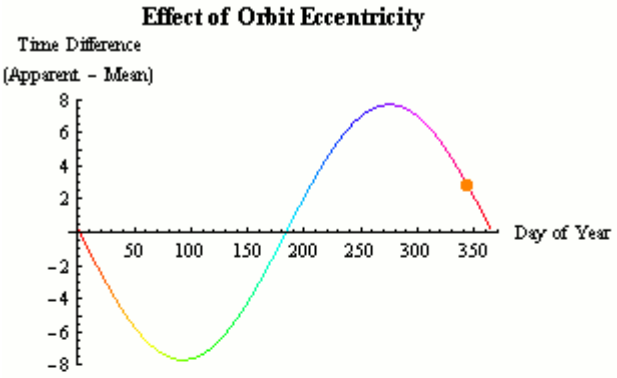
The analemma. Data courtesy of U.S. Coast and Geodetic Survey.

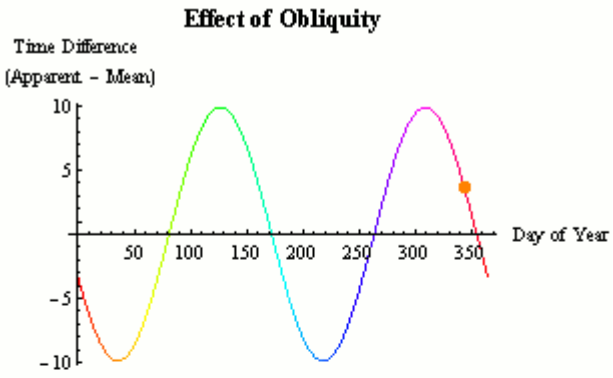
El Analema muestra que para todo sitio entre los trópicos de Cáncer y Capricornio (Lat. 23,45° N y 23,45° S) el Sol estará en posición cenital dos veces por año. En el trópico de Cáncer, en Junio 21. En el trópico de Capricornio, en Diciembre 22. El Sol nunca estará cenital fuera de los trópicos.

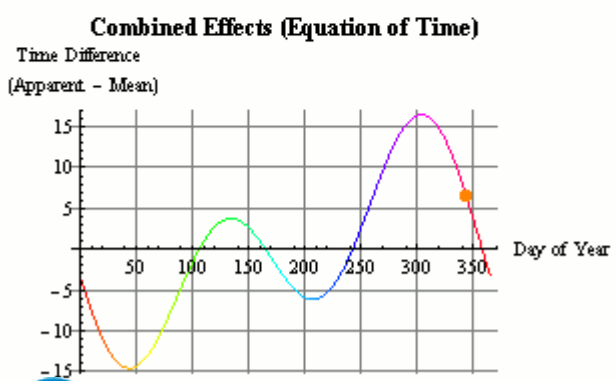
También muestra que el Sol estará adelantado al reloj un máximo de 16 m 33 seg en Noviembre 3, y retrasado un máximo de 14 m 6 seg en Febrero 12.

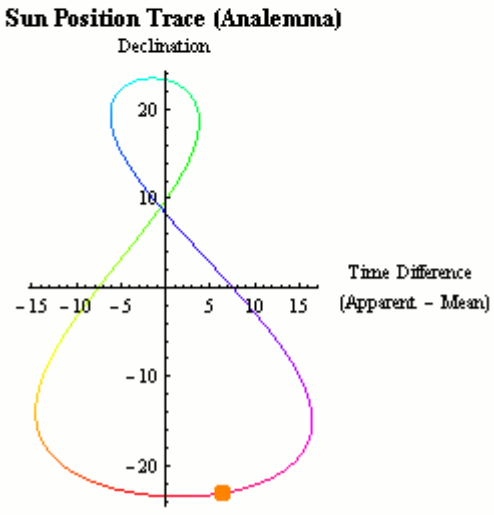
Este Analema fue preparado para la época 1950. Para la época 2000, la Ecuación del Tiempo es cero en Abril 15, Junio 13, Septiembre 1, y Diciembre 25, y tiene máximos en Mayo 14 y Noviembre 3, y mínimos en Febrero 11 y Julio 26.

La Ecuación del Tiempo adelanta 1 dia en 24,5 años. Ver Equation of time (Wikipedia, the free encyclopedia).







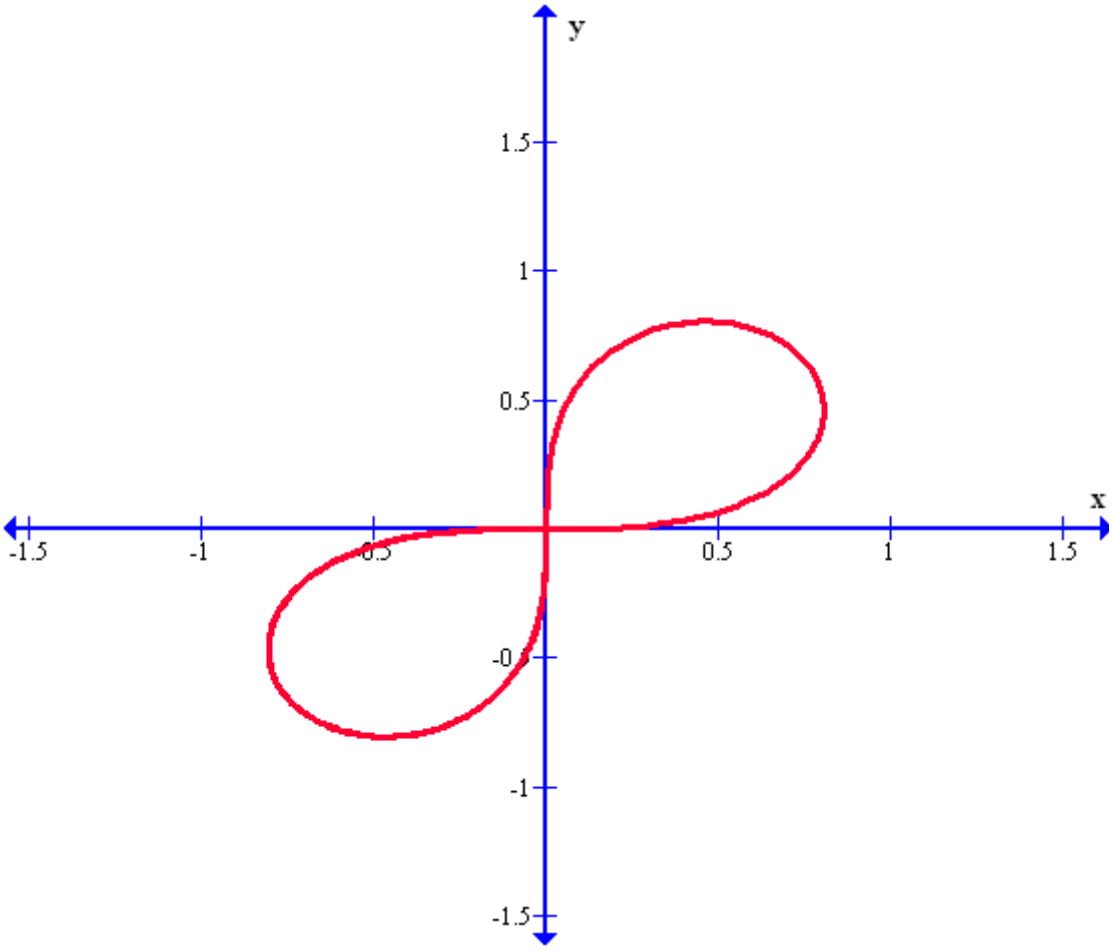


En matemáticas, una lemniscata es un tipo de curva descrita por la siguiente ecuación en coordenadas polares:

$$r^2 = a^2 \cos 2\,\theta$$

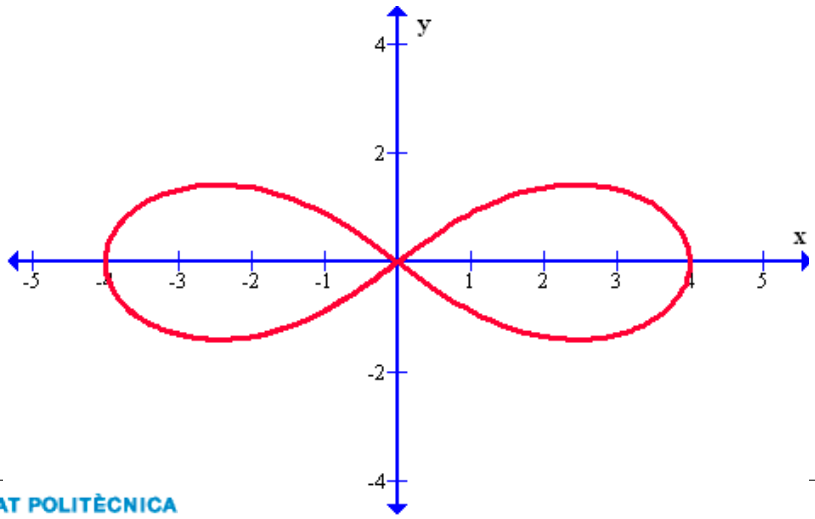
La representación gráfica de esta ecuación genera una curva similar al símbolo: ∞ , la curva se ha convertido en el símbolo del infinito y es ampliamente utilizada en matemáticas. El símbolo en sí mismo es, a veces, llamado lemniscata. Un ejemplo de esta función con su respectivo gráfico lo apreciamos a continuación:

r2 = sen 2 θ



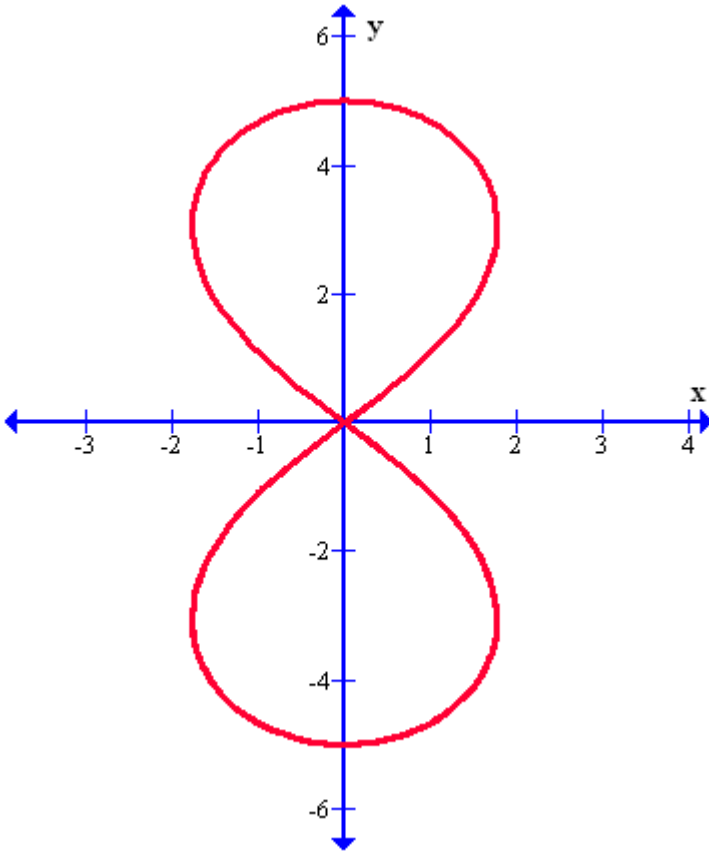
lemniscata a lo largo del eje x o en sentido horizontal:

r2 = 16 cos 2 θ



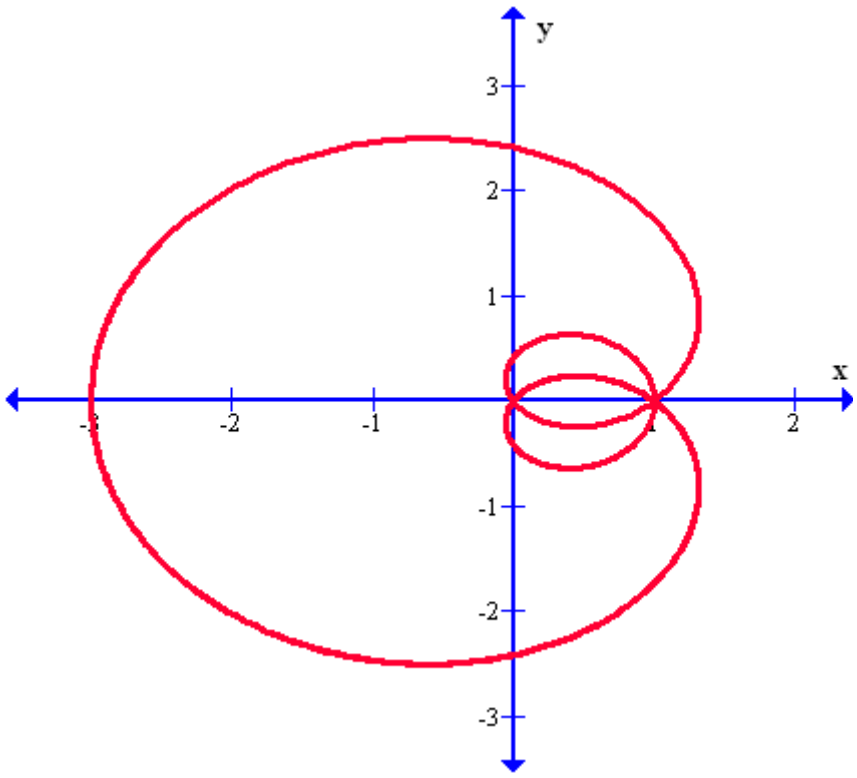
Tercer ejemplo una lemniscata a lo largo del eje Y en sentido vertical.

$r2 = -25 \cos 2\theta$



La nefroide de Freeth

Esta es una curva muy reciente si hablamos relativamente a las demás. Hay curvas polares que tienen varios siglos de existir, mientras que esta que trataremos en este momento es bastante reciente, pues fue desarrollada por el matemático inglés T.J. Freeth, quien descubrió esta curva en 1879. Un ejemplo se aprecia en este gráfico:

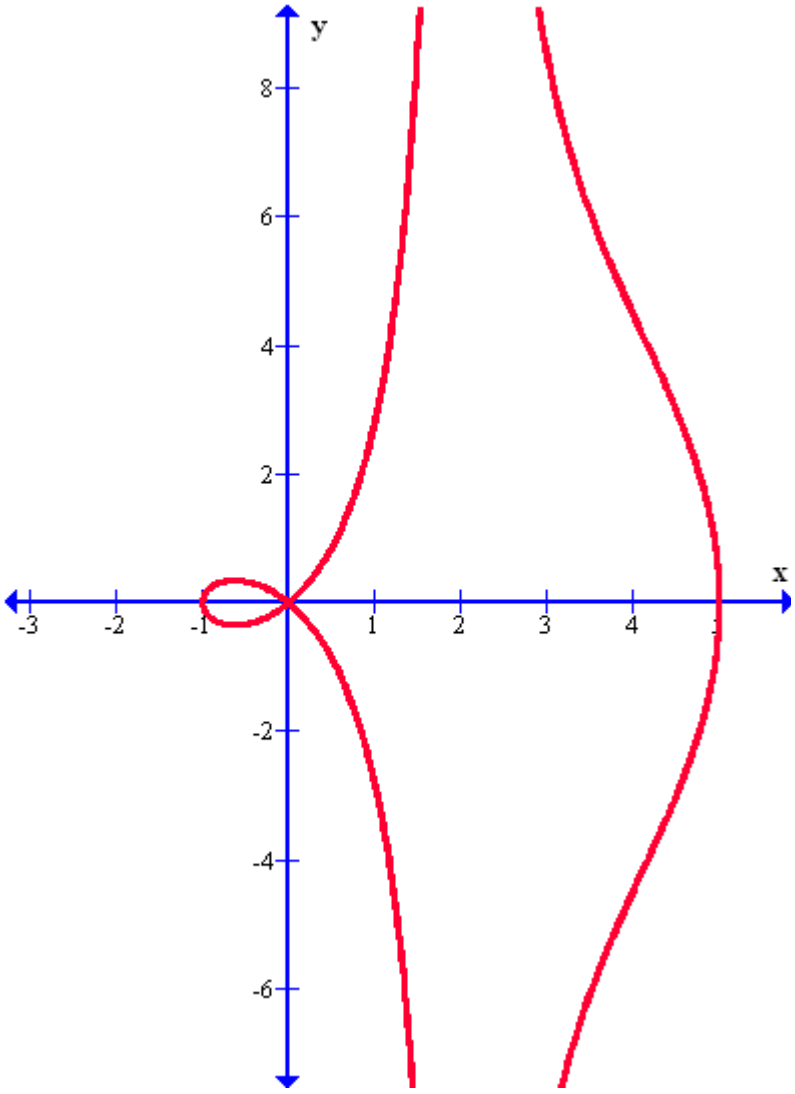


$$r = 1 + 2 \operatorname{sen} \frac{\theta}{2}$$

Concoides de Nicómenes

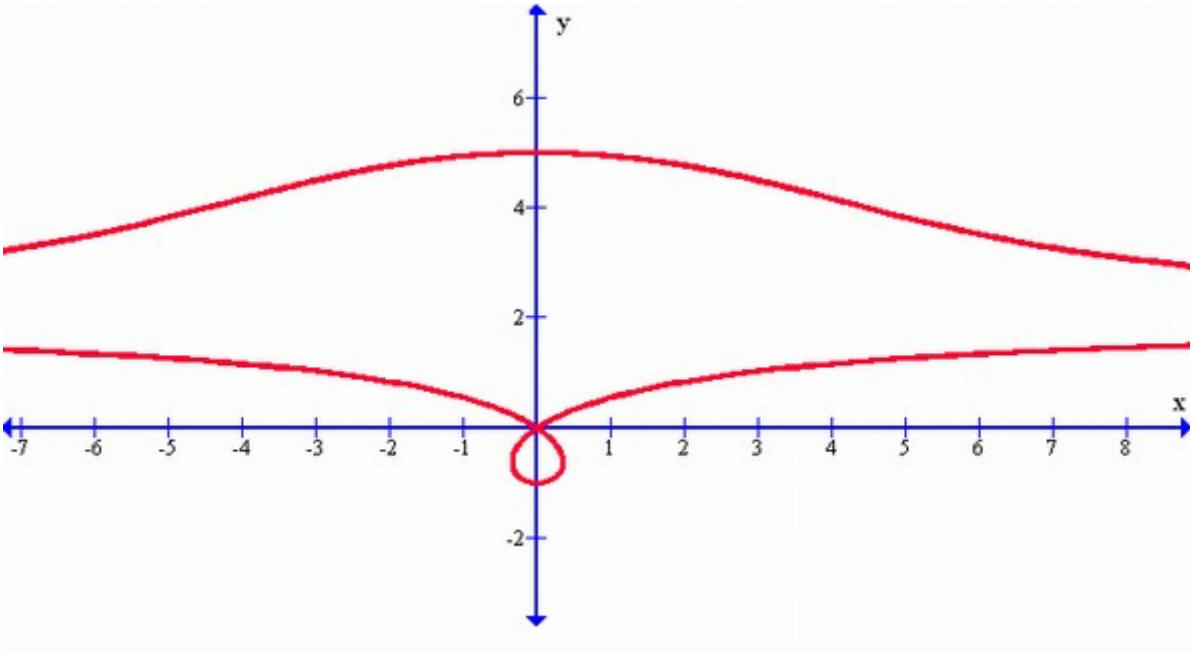
Nicómenes nació sobre el año 280 antes de Cristo en Grecia y murió en el año 210 a.C. Se sabe muy poco de su vida pero es famoso por su "Las líneas de la Concoide". Veamos un gráfico en coordenadas polares de la concoide de Nicómenes:

$$r = 2 \sec \theta + 3$$

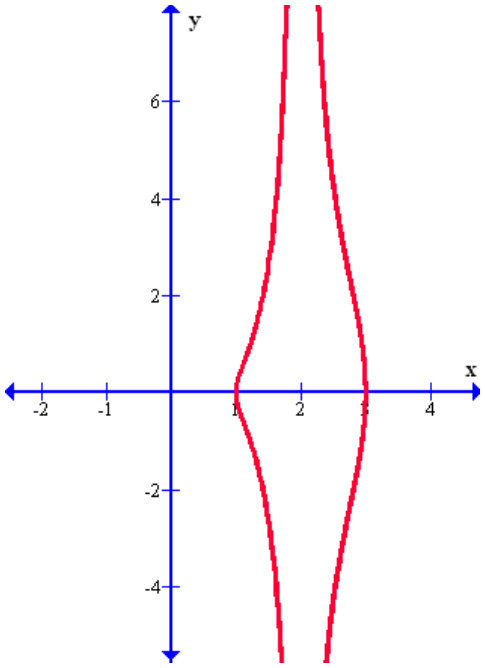


Veamos un nuevo ejemplo de una concoide de Nicómenes. La gráfica anterior está hacia la derecha, mientras que la que se presenta a continuación tiene una dirección hacia arriba. Veamos:

$$r = 2 \csc \theta + 3$$

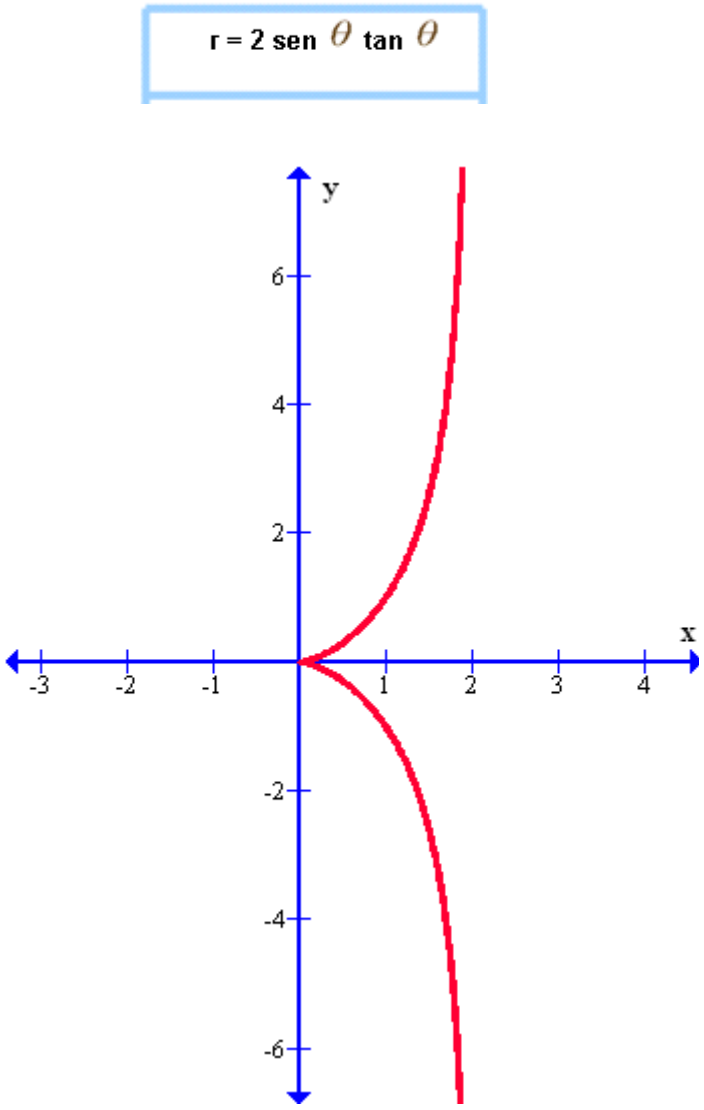


Un tercer ejemplo de Concoide de Nocómenes lo tenemos en el gráfico que se muestra a continuación, donde su forma se ve diferente a los dos gráficos anteriores de este mismo tipo debido a que se le está restando un número uno a la función. El mismo gráfico veríamos si se le estuviera sumando uno a la función. El gráfico quedará así:



Cisoude de Diocles

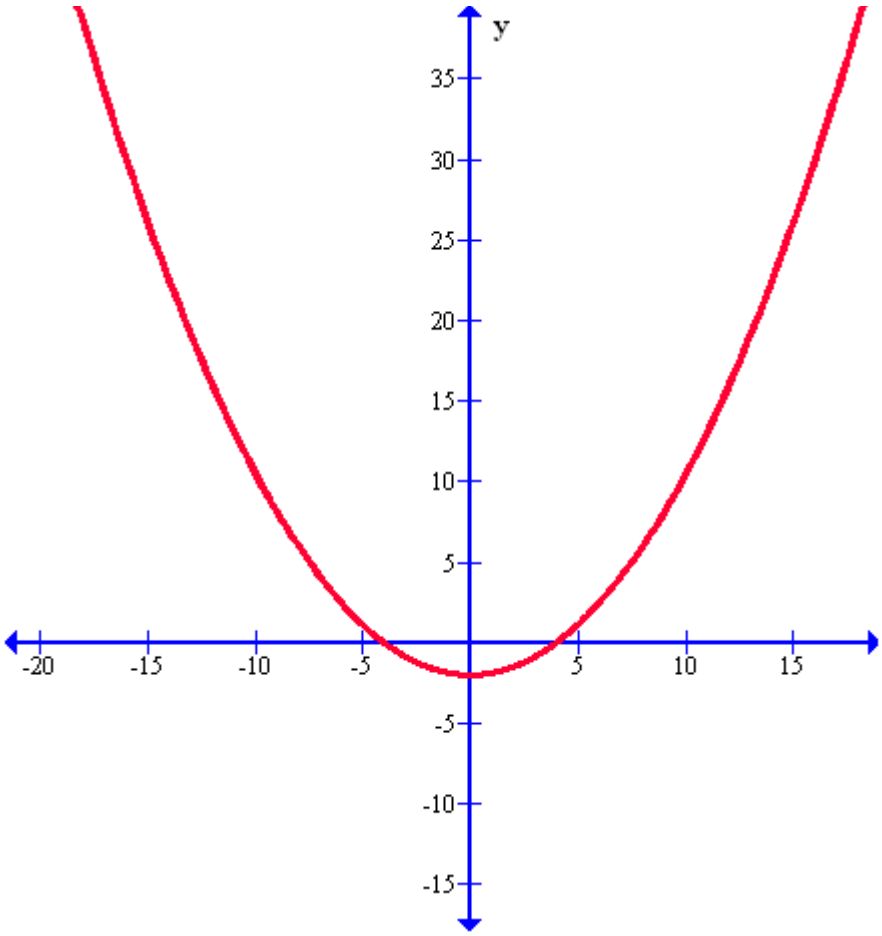
Esta es una curva muy famosa y útil en el cálculo. Fue utilizada por un matemático griego llamado Diocles para resolver el problema de la duplicación del cubo. El gráfico aparece de esta forma:



La parábola

Esta figura es muy conocida en el mundo del Cálculo. Tal como podemos generar funciones de parábolas en coordenadas cartesianas, lo podemos hacer también en coordenadas polares. Veamos el ejemplo:

$$r = \frac{8}{2 - 2\,\mathrm{sen}\,\theta}$$



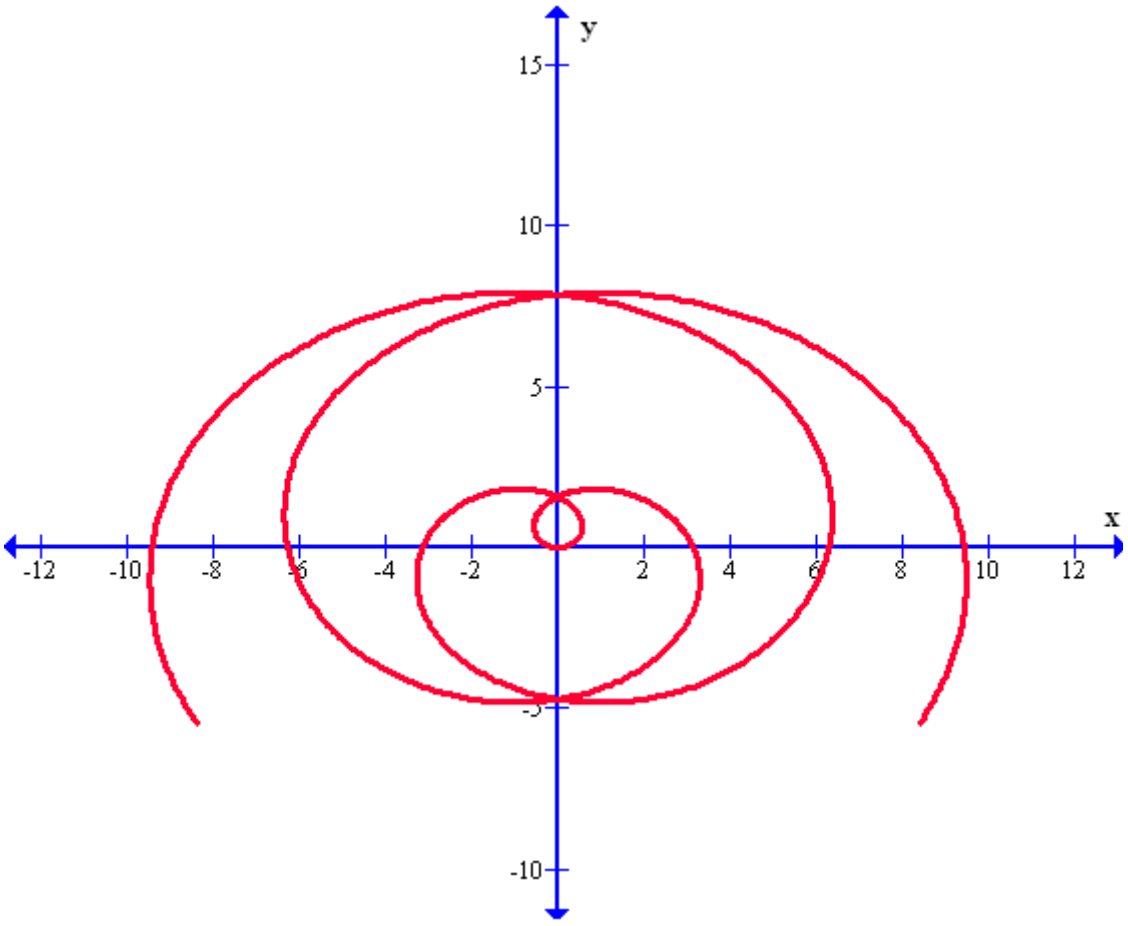
La espiral

Este gráfico tiene la forma de una espiral, tal como su nombre lo indica. La espiral más simple la podemos encontrar al observar una cuerda enrollada sobre sí misma. La forma de una espiral la vemos en una serpiente enrollada por ejemplo.

El gráfico que se presenta a continuación es también conocido como **Espiral de Arquímides** precisamente en honor Arquímedes, quien fue un notable físico y matemático griego que al ser fascinado por la belleza de esta curva, realizó un estudio profundo sobre sus propiedades matemáticas en su escrito titulado **Sobre las espirales,** escrito en el siglo III antes de Cristo.

Para mostrar el gráfico que se forma, presentamos la siguiente función en coordenadas polares que formará la espiral polar siguiente:

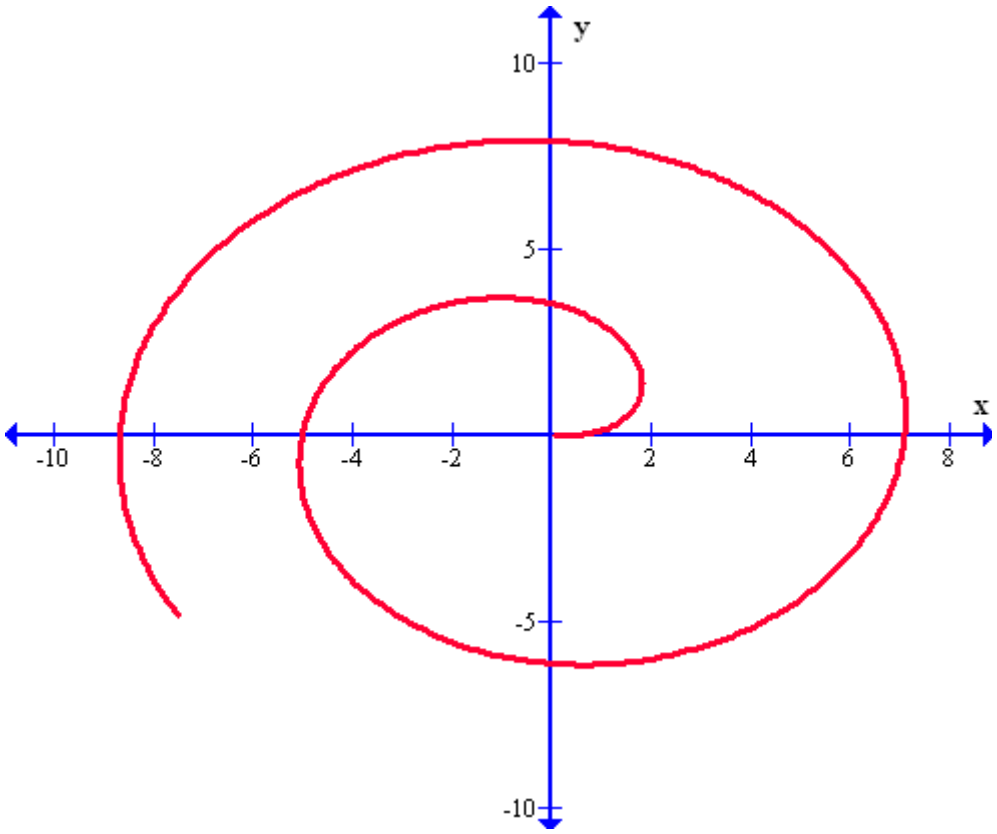
$$r = \theta$$



Veamos ahora otra gráfica espiral conocida como **espiral de Fermat**, pues fue examinada por Fermat en 1936. Su ecuación es $\mathbf{r^2 = a^2 + \theta}$.

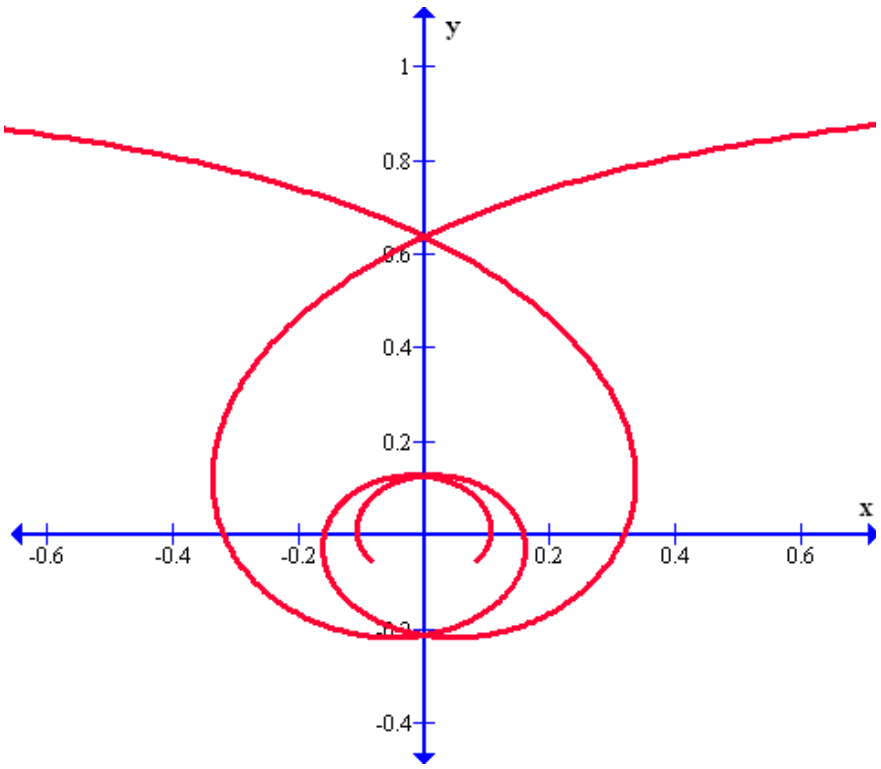
En el siguiente ejemplo se muestra una función y su respectiva gráfica que nos permiten conocer la **espiral de Fermat:**

$$r^2 = 8\theta$$



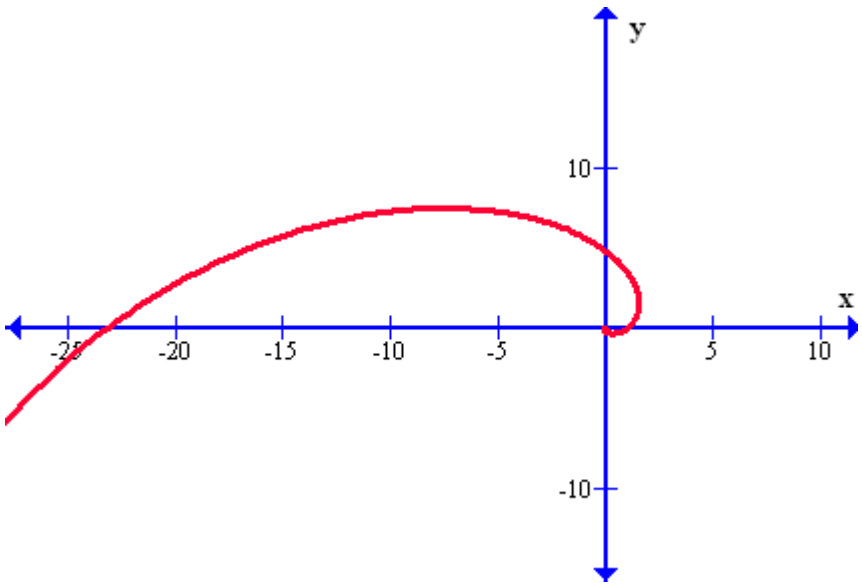
Un segundo gráfico espiral lo tenemos en la función que veremos ahora, que podríamos encontrarla con dos nombres refiriéndose al mismo gráfico. Ambos nombres equivalen a lo mismo como podremos apreciar . Dichos nombres con los que se conoce a esta espiral son: espiral recíproca o espiral hiperbólica. Tendremos entonces:

$$r = \frac{1}{\theta}$$



Otro caso que se
puede dar es la espiral logarítmica, que se ilustra mediante la siguiente función y su respectivo gráfico:

$$r = e^{\theta}$$



La evolvente

La **evolvente del círculo**, a veces llamada *involuta*, es una curva plana de desarrollo, cuyas normales son tangentes de la circunferencia.

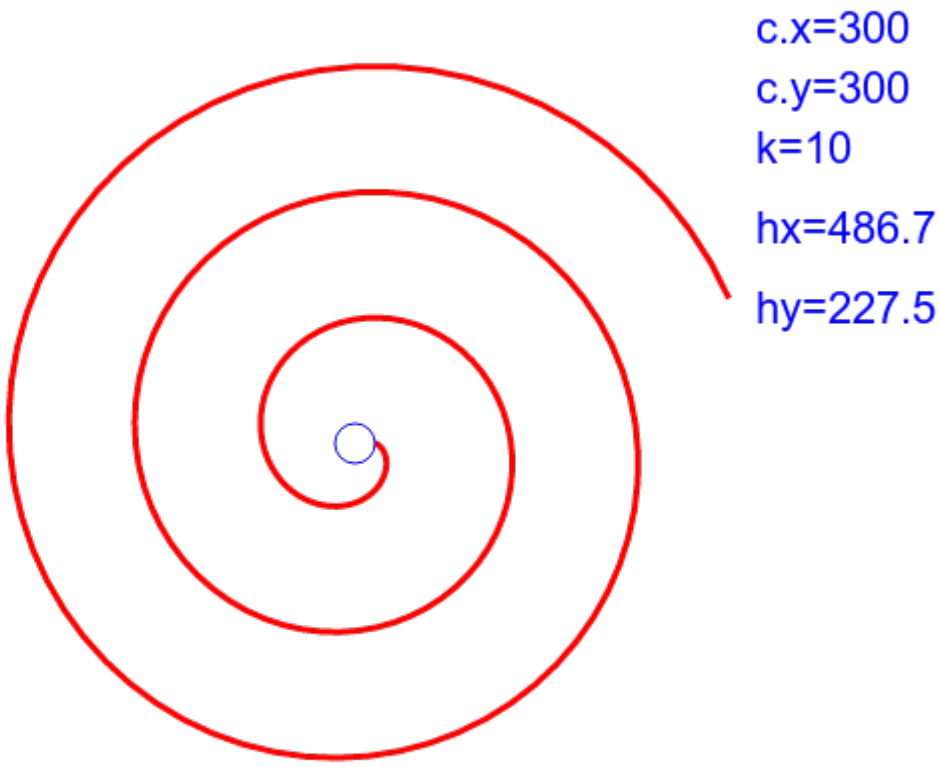
A menudo se traza sin saberlo: cuando un hilo tenso o un cable se desenrollan de una bobina circular sus puntos describen la evolvente de la circunferencia de esa bobina.

Fue estudiada originalmente por Christian Huygens,1 que trataba de diseñar relojes de péndulo para uso marino. Huygens utilizó la cicloide para forzar la oscilación regular del péndulo. Cuando un hilo tenso se enrolla en una cicloide cada uno de sus puntos describe un cicloide, es decir, la curva de desarrollo de una cicloide es una cicloide, como la de una circunferencia es una evolvente. La aplicación a los perfiles de las ruedas dentadas fue propuesta por Leonhard Euler

La curva se puede definir paramétricamente mediante la siguiente ecuación:

$$x(t) = k(cost + t\,sent)$$
$$x(t) = k(sen\,t - t\,cost)$$

donde k es el radio en el punto de contacto.



La librería Traza.js ,(Recapitulando)

He creado la librería Traza.js que encontrareis dentro de la carpeta scripts, para facilitar los procesos de dibujo de lineas,puntos y uniones entre estos.
Para explicarme permitirme recapitular sobre estos términos:

- Debemos crear con HTML5 un esqueleto o 'framework' donde vamos a incrustar nuestro código javascript.

Este sin todavía el código HTML5 sera así:

```
<html>
<head>
</head>
<body>
</body>
<html>
```

- Ahora entre las etiquetas <body> y </body> insertamos nuestras etiquetas <script></script>

por que lo primero que se carga en un documento HTML es lo que hayamos declarado en el <body> o cuerpo.

Ahora se vera así:

```
<html>
<head>
</head>
<body>
<script>

//aqui va el codigo



<script>
</body>
```

- Ahora vamos a declarar el canvas o entorno gráfico de javascript reconocido por cualquier navegador para poder dibujar.

```
<html>
<head>
</head>
<body>
</body>
<html>

<canvas id="canvas" width="800" height="622" style="border:1px solid grey;
float:left; margin-right:10px;">

</canvas>
<script>
// aquí el codigo
</script>
</body>
</html>
```

- Como veis el canvas tiene un **id** o referencia dentro del documento unas **propiedades**:

(Height ,width),un **estilo** color,borde, fondo,etc. Vamos por ejemplo ha introducir un código que dibuje dos puntos y una linea que los unen.

```
<html>
<head>
</head>
<body>
</body>
<html>

<canvas id="canvas" width="800" height="622" style="border:1px solid grey;
float:left; margin-right:10px;">

</canvas>
<script>

document.getElementById("canvas");
ctx=c1.getContext("2d");
```

```
</script>
</body>
</html>
```

Las dos primeras lineas introducidas en el código entre las etiquetas <script> y </script>son indispensables, tanto es así que figuraran en todos nuestros códigos por que con ellas estamos referenciando,o lo que es lo mismo diciéndole al navegador que estamos utilizando en primer lugar, el lenguaje de programación **javascript** y que también estamos utilizando su entorno gráfico **canvas.**

Podemos cambiar del **canvas** ,su **id** en vez de 'canvas' podemos llamarle el nombre que queramos (entre comillas), sus propiedades y estilos.

```
<html>
<head>
</head>
<body>

<canvas id="canvas" width="800" height="622" style="border:1px solid grey;
float:left; margin-right:10px;">

</canvas>
<script>

document.getElementById("canvas");
ctx=c1.getContext("2d");

//dibujamos dos puntos y una linea que los une



//primer punto


ctx.moveTo(300,300);
ctx.arc(300,300,5,0,Math.PI*2,false);
```

```
//segundo punto

ctx.moveTo(100,300);
ctx.arc(100,300,5,0,Math.PI*2,false);



// dibujamos la primera linea
ctx.moveTo(300,300);
ctx.LineTo(100,300);
ctx.stroke()


</script>
</body>
</html>
```

como veis es tedioso repetir siempre **ctx** que es el contexto que le damos al canvas,**ctx.moveTo(x,y)**

ctx.LineTo(x,y) y dibujar un punto seria **ctx.arc(Centrox,Centroy,radio,inicio en radianes,final en radianes,sentido del giro).**

Por eso con la Libreria Traza.js nos ahorramos el trabajo de repetir estos comandos, echemos un vistazo al código:

```
function TrazaCamino(ctx,array,grosor,color,color2){
ctx.save();
ctx.fillStyle=color2;
ctx.lineWidth=grosor;
ctx.strokeStyle=color;
ctx.beginPath();
ctx.moveTo(array[0].x,array[0].y);
for(var i=1;i<array.length;i++){
ctx.lineTo(array[i].x,array[i].y)
}
ctx.fill();
ctx.stroke();
ctx.restore();
```

```
}


function PuntosArray(ctx,array,grosor,r,color1,color2,inc){

ctx.save();
ctx.lineWidth=grosor;
ctx.strokeStyle=color1;
ctx.fillStyle=color2;
ctx.beginPath();

for(var i=0;i<array.length;i+=inc){
ctx.moveTo(array[i].x,array[i].y)
ctx.arc(array[i].x,array[i].y,r,0,2*Math.PI,false);
}

ctx.stroke()
ctx.fill();
ctx.restore();

}
```

```
function TrazaCamino(ctx,array,grosor,color,color2){
```

recibe 5 parámetros :

ctx; el contexto del **canvas**

Array:*un vector que contendrá una serie de* ***objetos****, todos los puntos a unir:*

({x:x0,y:y0},{x:x1,y:y1},{x:x3,y:y4}.....)

grosor:grosor de la linea.

Color:color del trazado.

color2:color del fondo de la figura resultante de unir varios puntos con rectas por ejemplo un poligono.

Si el significado **objetos** no tiene sentido para usted remítase a la pagina 6,por que es de vital importancia ya que esta programación se basa en **objetos**,

```
function PuntosArray(ctx,array,grosor,r,color1,color2,inc){
```

esta función es básicamente lo mismo pero en vez de rectas traza puntos.

Como se puede ver ambas funciones nos ahorraran escribir y repetir muchas lineas de codigo con tan solo invocarlas y pasar-les los argumentos que queramos.

Debemos anadir Traza.js como fichero externo en nuestro documento veamos como se hace:

```
<script text=text/javascript src='js/Traza"></script>
```

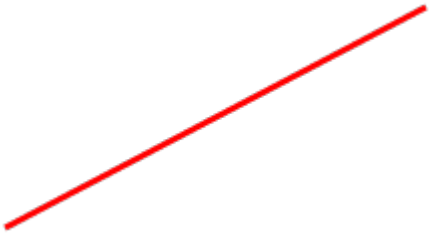
En este caso Traza.js se encuentra dentro una carpeta llamada 'js' si hace falta declaramos una ruta donde tengamos guardado el fichero

Esto quedara:

```
<html>
<head>
<script text=text/javascript src='js/Traza"></script>
</head>
<body>
</body>
<html>

<canvas id="canvas" width="800" height="622" style="border:1px solid grey;
float:left; margin-right:10px;">

</canvas>
<script>
document.getElementById("canvas");
ctx=c1.getContext("2d");
//invocamos a la función TrazaCamino y le pasamos los parametros

TrazaCamino(ctx,[{x:100,y:300},{x:300,y:300}],1,'red','red')

</script>
```

```
</body>
</html>
```

Y este seria el resultado de nuestro código:



No tese que el array de objetos va entre corchetes **[{ x:100, y: 300} , { x:300, y: 300 }]** y contiene dos objetos representando las coordenadas de los dos puntos referenciados,un array o vector puede también declararse en el encabezamiento del código como por ejemplo queremos que nuestro array se llame 'puntos' asi pues:

var puntos=new Array();

otra manera es simplemente

var puntos=[];

La palabra **var** es una palabra reservada que quiere decir variable

así a nuestra función TrazaCamino le pasamos el array **puntos** ya declarado

```
TrazaCamino(ctx, [ {x:100, y: 300 } , { x:300, y:300 } ], 1, 'red', 'red')
```

o bien introducimos los puntos en un array uno por uno por que la función 'espera ' un array como

su segundo argumento.

```
TrazaCamino(ctx, puntos, 1 ,'red', 'red' )
```

tenemos estas dos opciones tanto una como otra es valida.

Por lo tanto la utilización de Traza.js hará nuestro código mas legible e inteligible.

Es muy importante:

- cerrar todas las etiquetas que se abren <script> <script/> en todo el código

tanto en canvas ,HTML o java script;

- Todos estos códigos,los lenguajes HTML,javascript,css3,son interpretados y ejecutados por el navegador,por lo que no necesitamos bajar ningún programa ni entorno ni compiladores de programación, nos bastara un simple editor de textos y guardarlos con la extensión .html

Un ejemplo de programación aplicado al ámbito Náutico: Un simulador G.M.D.S.S 3D

El G.M.D.S.S es un sistema que engloba una serie de protocolos y procedimientos destinados a preservar la seguridad de la vida humana en el mar .

El propósito de este sistema: alertar rápidamente a las autoridades encargadas de busqueda y salvamento, así como también a otros buques que se encuentren en las cercanías del buque siniestrado,la demora:la mínima.

El sistema GMDSS divide todos los mares en cuatro áreas de navegación. Según el área en el cual opera el buque deberá llevar a bordo algunos o todos los equipos del sistema GMDSS. Zona A1 Es aquella zona que esta bajo la cobertura de una estación en tierra que posea cobertura total en VHF LSD canal 70.

Zona A2 Es aquella zona que esta bajo la cobertura de una estacion en tierra que posea cobertura total en MF LSD en la frecuencia de socorro de 2.187,5 Khz.

Zona A3 Es aquella zona que esta bajo la cobertura de una estación en tierra que posea cobertura total de los satélites de comunicaciones INMARSAT. Aproximadamente entre los 70ª norte y los 70ª sur.

Zona A4 Es aquella zona que no esta comprendida en ninguna de la zonas anteriores. Actualmente seria por encima de los 70º norte y por debajo de los 70º sur (zonas polares)."

el simulador SPR programado con java script .

Con este simulador vamos ha realizar una llamada de socorro digital lo que nos llevara ha comprender como navegar por las opciones de su menú.

1. Pulsamos la tecla POWER.

Se nos activara el mensaje de bienvenida ,y tras ello aparecerá en pantalla nuestra posición ,la hora G.T y los canales de recepción/transmisión digitales.

Podemos cambiar los parámetros de opacidad de la pantalla con la tecla **Dim** al pulsarla nos cambiara el menú a una pagina con dos flechas una para subir y otra para bajar los valores de la opacidad o transparencia de la pantalla.

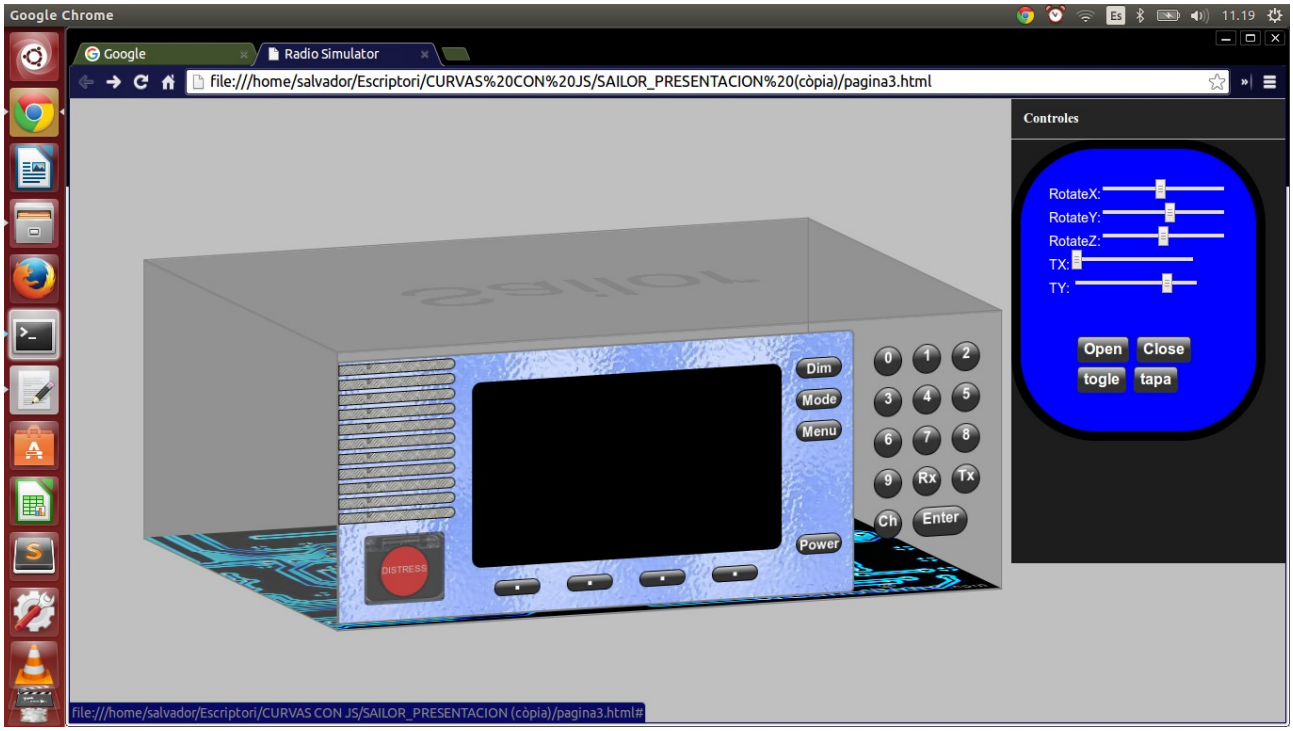
2. Pulsamos la tecla **menú** y escogemos la primera opción **DSC CALL** y pulsamos **Enter** nos aparecera un nuevo menú en el cual escogeremos **Alert** volvemos a pulsar **Enter** y escogemos **Nat:Cause ofDistres => Enter a**qui podemos escoger entre todas las causas que han originado esta situacción de peligro: fire,capsizing,aground piracy atack,listing,explotion,seleccione por ejemplo la primera **fire al darle a Enter** aparecera la pantalla con su posicion naturaleza del peligro tipo de llamada en este caso dirigida a todas las estaciones y a todos los barcos que nevegen por su zona.

Al pulsar nuevamente **Enter** aparecerá en la pantalla un mensaje que le indica que pulse el botón rojo de **Distresss**

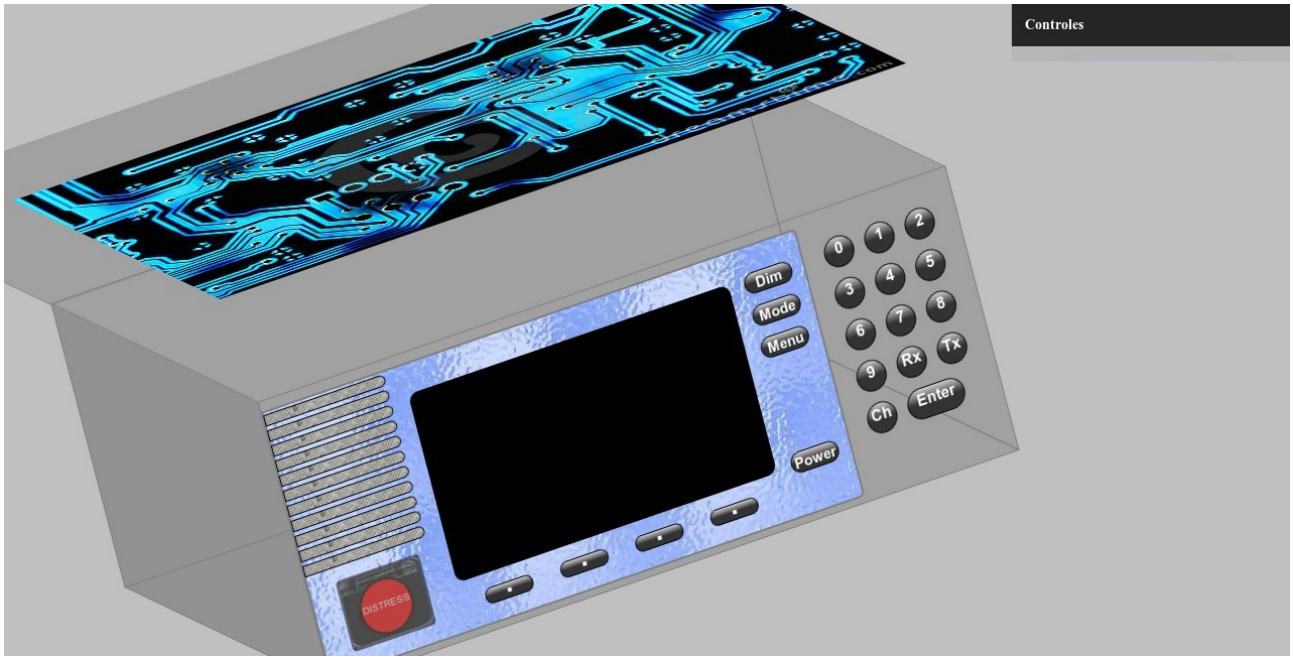
Al pulsar el botón rojo de **Distresss** su llamada sera retransmitida automáticamente cada 4 segundos hasta ser recibida por alguna estación o barco que vendrá en su auxilio.

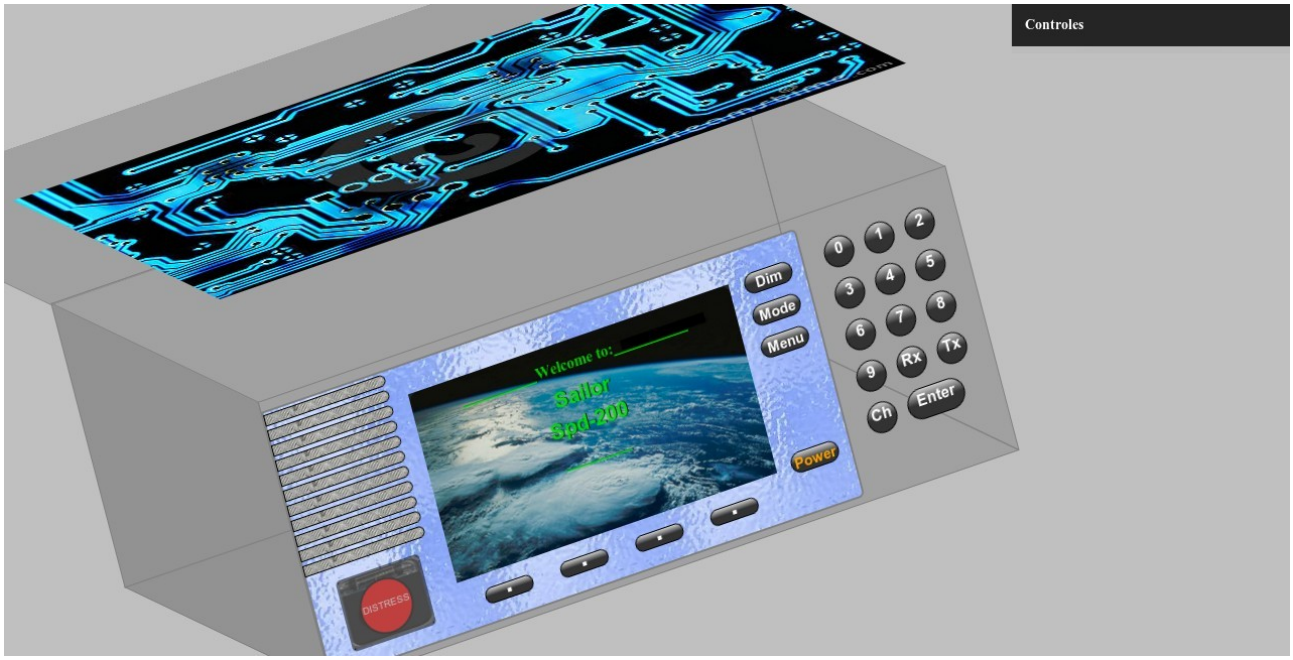
Barra de controles

El simulador S.P.R. Sailor posee una barra de controles que aparece en el margen superior derecho y que se expande al posar el cursor sobre el. Con estos controles deslizantes y botones podemos variar su posición respecto a un sistema de coordenadas 3D respecto a los ejes X,Y, Z.
Que serán sus parámetros respecto a una matriz de transformación ,que implica rotaciones y traslaciones respecto a los ejes X,Y,Z .



el simulador SPR Sailor y su barra de controles.





el simulador SPR Sailor en distintas configuraciones de su matriz de transformación..

Código del S.P.R Sailor

```
<html>
<head>
<title>Radio Simulator</title>
<link href="css2/basic.css" rel="stylesheet" type="text/css" media="all">
<link href="css2/controles.css" rel="stylesheet" type="text/css" media="all">
<link href="css2/pantallas4.css" rel="stylesheet" type="text/css" media="all">
<script type="text/javascript" src="js/reloj.js"></script>
<script type="text/javascript" src="js/barra.js"></script>

<style>

body {
overflow:hidden;
}

.view {
position:absolute; left:300; top:0;
-webkit-perspective:0px;
-webkit-transform-style: preserve-3d;
-webKit-Transform:scale(0.1);

}
.face {
background:rgba(130, 130, 130, 0.5); //-webkit-radial-gradient(white,beige 20%,
beige 60%, black);
border: 1px solid grey;
height:300px; width:750px;
position:absolute; left:-80px; top:30px;
-webkit-backface-visibility :visible;
}

.face2 {
background:rgba(130, 130, 130, 0.5);//-webkit-radial-gradient(white,beige 20%,
beige 60%, black);
border: 1px solid grey;
height:600px;
width:750px;
position:absolute;
left:0px;
```

```
top:0px;
}

.esquinas {
background:rgba(130, 130, 130, 0.5);
//-webkit-radial-gradient(white,beige 20%, beige 60%, black);
border: 1px solid grey;
height:300px; width:600px;
position:absolute; left:-80px; top:30px;
}

.face_principal {
 background:transparent;
-webkit-radial-gradient(white,beige 20%, beige 60%, black);
border: 0px solid grey;
height:300px; width:750px;
position:absolute; left:-80px; top:30px;
-webkit-backface-visibility :visible;
}

#top {
-webkit-transform-origin: 50% 100%;
-webkit-transition: -webkit-transform 2s;
height:600px; width:750px;
color:rgba(130, 130, 130, 0.8);
font-size:155px;
text-align:center;
line-height: 210px;
}

#circuito {
-webkit-transition: -webkit-transform 3s;
z-index:5;
}

.axis {
font: 18pt helvetica bold;
 background: -webkit-linear-gradient(top left,rgba(0,0,0,0.5),#f0f0f0);
position: absolute; left: -280px; top: 0px; width:500;
border:1px solid grey;
}
```

```
p { padding:28;}


#cont{
 position:relative;
width:300px;
height:300px;
left:30px;
top:30px;
 color:white;
 line-height: 0px;
 z-index:10;
}

#controles{
position:absolute;
top:0px,
left:left 100% ;
width:250px;
height:300px;
background:blue;
border: 10px solid black;
-webkit-border-radius: 90px;
}


b{font-family:tahoma;}

span{color:red;}

#distress{
position:absolute;
left:30px;
top:190px;
width:87px;
height:71px;
border:0px solid orange;

}
#tapa{
```

```
	position:absolute;
	left:28px;
	top:188px;
	width:87px;
	height:71px;
	border:2px solid black;
	z-index:1;
	background:grey;
	opacity:0.5;
	-webkit-border-radius:5px;
	}
	#boton_rojo{

	position:absolute;
	left:45px;
	top:200px;
	width:53px;
	height:53px;
	border:2px solid black;
	}

	.deg2 {

	line-height:50px;
	background-image: -webkit-gradient( linear, left top, left bottom, color-
stop(0.01,red),
	color-stop(0.5,red) );
	border: 1px solid #333;
	border-radius: 4px 4px 4px 4px;
	color: #FFFFFF;
	font-size: 10px;
	-webkit-border-radius:360px;
	text-align: center;
	}


	#cuerpo{

	border: 1px solid grey;
	background:transparent;
```

```
-webkit-border-radius: 0px;

}


#cara_frontal{

border: 2px solid grey;
background:url(img/metales/1.jpg)  ;

}

#ralla1{ background:url(img/metales/4.jpg);}
#ralla2{ background:url(img/metales/4.jpg);}
#ralla3{ background:url(img/metales/4.jpg);}
#ralla4{ background:url(img/metales/4.jpg);}
#ralla5{ background:url(img/metales/4.jpg);}

#ralla6{ background:url(img/metales/4.jpg);}
#ralla7{ background:url(img/metales/4.jpg);}
#ralla8{ background:url(img/metales/4.jpg);}
#ralla9{ background:url(img/metales/4.jpg);}
#ralla10{ background:url(img/metales/4.jpg);}
#ralla11{ background:url(img/metales/4.jpg);}
#ralla12{ background:url(img/metales/4.jpg);}

h3{color:lime; font-size:16px; line-height:-10px;}



.mensaje{
 position:relative;
width:200px;
height:50px;
left:30px;
top:30px;
 color:red;
 line-height: 0px;
 z-index:10;
```

```
}


#capa2{
position:absolute;
width:400px;
height:50px;
left:700px;
top:430px;
 color:red;
 line-height: 0px;
 z-index:1000;
border:4px solid black;
background:red;


}

</style>
```

Se empieza por la etiqueta <html> luego abrimos con la etiqueta <head> y entre las etiquetas <head> y </head> añadimos las rutas de los ficheros externos de estilos y javascript con sus respectivas extensiones css y js.

```
<title>Radio Simulator</title>
<link href="css2/basic.css" rel="stylesheet" type="text/css" media="all">
<link href="css2/controles.css" rel="stylesheet" type="text/css" media="all">
<link href="css2/pantallas4.css" rel="stylesheet" type="text/css" media="all">
<script type="text/javascript" src="js/reloj.js"></script>
<script type="text/javascript" src="js/barra.js"></script>
```

Incluimos después dentro de las etiquetas **<style>** y **</style>** los estilos básicos que configurarán el simulador y las imágenes que darán textura a su cuerpo que estaran en la carpeta de '**img**' .

Después incluimos entre las etiquetas <script> y </script> nuestro código java script.

```
<script type="text/javascript">
var crece=true;
var encendido=false;
var pantalla_actual="menu";
var brillo=0;
var num_pantallas=5;
var fila_actual=1;
var audio;
var con=0;
var letras;
var inc=0;
var but_arrows="";
var bucle_welcome;
var bucle;
var bucle3;
var but_cancel="status";
var TextoAct="";
var TextoAct2="";
var n=-1;
var n2=-1;
var n3=-1;
var fila;
var option_selected;
var max;
var name;

function init()
{

hidden_all();
setXY();
var texto=document.getElementById('texto')
texto.innerHTML=1;
}

function power()
```

```
{

var welcome=document.getElementById('pantalla_welcome');
var power=document.getElementById('power');

 welcome.style.opacity=0.9;

audio = document.createElement("audio");
 audio.src = "audio/bell";
 audio.play();
reloj();
barra();
(!encendido)?encender("welcome"):apagar();
}

function encender(pantalla_actual){


var power=document.getElementById('power');
encendido=true;
power.style.color='orange';
if(pantalla_actual=="welcome") animar();
```

```
visualiza(pantalla_actual);


}



function apagar(){

hidden_all();
```

```
var power=document.getElementById('power');
clearTimeout(bucle_welcome);
```

```
clearTimeout(bucle);
power.style.color='white';
encendido=false;
}
function hidden_all(){
var pantallas=document.getElementsByClassName('pantalla');
for (i=0;i<pantallas.length;i++) { pantallas[i].style.visibility="hidden"; }
 document.getElementById('cont_barra').style.visibility="hidden";

}


function visualiza(option) {


if(encendido){

//alert("mostra pantalla_"+option);
//--------------------------reloj-----------------------------_//
var hora=document.getElementById('reloj');
//--------------------------BARRA-----------------------------------_//

var cont_bar =document.getElementById('cont_barra');
var bar =document.getElementById('barra');

hidden_all();

if (option.substr(0,8)=="1_4_1_3_") option="info"; // cas especial
if (option=="call") {document.getElementById('capa22').style.visibility='visible';}

but_cancel=document.getElementById('pantalla_'+option).getAttribute("cancel");
but_arrows=document.getElementById('pantalla_'+option).getAttribute("arrows");

option_selected=document.getElementById('pantalla_'+option).getAttribute("default")
;
document.getElementById('pantalla_'+option).style.visibility="visible";

fila_actual=1;
max=document.getElementById('pantalla_'+option).getAttribute("max");
name=document.getElementById('pantalla_'+option).getAttribute("name");
if (but_arrows=="select_menu")
```

```
document.getElementById("fila"+option_selected).setAttribute("class","fila
selected");
        audio = document.createElement("audio");
        audio.src = "audio/bell.au";
        audio.play();

      if (!encendido||option==undefined) return false;

      //visualizar reloj y barra
      if(document.getElementById('pantalla_status').style.visibility=="visible")
{hora.style.visibility="visible";
      bar.style.visibility="visible";cont_bar.style.visibility="visible";}

      else
{hora.style.visibility="hidden";bar.style.visibility="hidden";cont_bar.style.visibility="
hidden";}



      }


      }



      function animar()
      {

      letras=new Array();
      var texto="<h3>"+"_________"+"<p3>"+"Welcome
to:"+"</p3>"+"_________"+"</h3>"+"<h2>"+"Sailor"+"<br><br>"+"Spd-
200"+"</h2>"+"<h3>"+"_________"+"<br><br>"+""+"</h3>"
      for(i=0;i<texto.length;i++)
      {
      letras[i]=texto.charAt(i);
      }
      mueveLetras();

      }
```

```
function mueveLetras()
{

n++;
TextoAct += letras[n];
var capa = document.getElementById("capa").innerHTML="<h1>"+TextoAct;
document.getElementById("capa").style.fontSize=7
bucle_welcome=setTimeout("mueveLetras()",50);
if(n==letras.length-1)
{
        n=-1;
        TextoAct="";
        clearTimeout(bucle_welcome);
        visualiza("status");
}

}

function animar2()
{
letras2=new Array();
var texto2="<h3>"+"PRES THE DISTRESS BOTTON";
for(i=0;i<texto2.length;i++)
{
        letras2[i]=texto2.charAt(i);
}
mueveLetras2();

}
```

```
function mueveLetras2()
{
n2++;
TextoAct2 += letras2[n2];
var capa2 =
document.getElementById("capa22").innerHTML="<h1>"+TextoAct2;
```

```
document.getElementById("capa22").style.fontSize=7
bucle=setTimeout("mueveLetras2()",50);
if(n2==letras2.length-1)
{
        n2=-1;
        TextoAct2="";
        //clearTimeout(bucle);
        //visualiza("otra");
}

}


function cancel() {
document.getElementById('capa22').style.visibility='hidden';
clearTimeout(bucle_welcome);visualiza(but_cancel);
}

function setvalue(action)
{

if (but_arrows=="brillo") {

        document.getElementById('tabla1').style.background='transparent';

        if(action=="up"&&brillo<1){ brillo=brillo+0.1 }
        if(action=="down"&&brillo>0.2){brillo=brillo-0.1}
```

```
        document.getElementById('pantalla_welcome').style.visibility='visible';
        document.getElementById('pantalla_welcome').style.opacity=1;
        var pantalla= document.getElementById('tabla1');
        var texto=document.getElementById('texto');
        texto.innerHTML=(brillo*10).toFixed(0);
        document.getElementById('tabla1').style.background='transparent';
        document.getElementById('pantalla_welcome').style.opacity=brillo;
        document.getElementById('capa').style.visibility='hidden';
```

```
            }

            else if (but_arrows=="select_menu") {
            if (fila_actual==undefined) fila_actual=1;
            if(action=="up"&&fila_actual<max) fila_actual++; else if
(action=="down"&&fila_actual>1) fila_actual--;
            option_selected=name+fila_actual;
            console.log(option_selected);
            var filas=document.getElementsByClassName('fila');
           for (i=0;i<filas.length;i++) { filas[i].setAttribute("class","fila"); }

document.getElementById("fila"+option_selected).setAttribute("class","fila
selected");



            }
      }



      function setXY() {
      animar2();
      var xRot = document.getElementById("xSlide").value;
      var yRot = document.getElementById("ySlide").value;
      var ZRot = document.getElementById("zSlide").value;
      var TX = document.getElementById("TXSlide").value;
      var TY = document.getElementById("TYSlide").value;
      var tString="rotateX("+xRot+"deg) rotateY("+yRot+"deg) rotateZ("+ZRot+"deg)
translateX("+TX+"px) translateY("+TY+"px)";
      document.getElementById("view").style.webkitTransform=tString;
      document.getElementById("view").style.webkitTransition=2+'s';
      }

      function openBox() {
      document.getElementById('top').style.webkitTransform = 'translateY(-600px)
rotateX(90deg)';
      document.getElementById('circuito').style.webkitTransform = 'translateY(-300px)
rotateX(90deg)';

      }
```

```
	function closeBox() {
	document.getElementById('top').style.webkitTransform = 'translateY(-600px)
rotateX(-90deg)';
	document.getElementById('circuito').style.webkitTransform = 'translateY(30px)
translateX(-80px) translateZ(300px) rotateX(90deg)';
	}

	function levanta() {
	document.getElementById("tapa").style.webkitTransition=2+'s';
	document.getElementById('tapa').style.webkitTransform = 'translateY(-71px)
rotateX(180deg)';
	}


	function baja_tapa() {
	document.getElementById("tapa").style.webkitTransition=2+'s';
	document.getElementById('tapa').style.webkitTransform = 'translateY(0px)
rotateX(0deg)';
	}



	function togleAxis() {
	var cusid_ele = document.getElementsByClassName('axis');
	for (var i = 0; i < cusid_ele.length; ++i) {
	var item = cusid_ele[i];
	(item.style.visibility=='hidden')?item.style.visibility=
'visible':item.style.visibility='hidden';
	}
	}
```

```
	function shake() {

	document.getElementById('view').style.webkitTransform = 'translateY(100px)
rotateZ(60deg) scale(0.001)';
	document.getElementById("view").style.webkitTransition=2+'s';
	setTimeout("cambia()",1500);

	}
```

```
function cambia() {

location.href="pagina2.html"
}
function distress() {
option='distress';
setTimeout("shake()",1000);
document.getElementById('menu').style.visibility='hidden';
visualiza(option);
}
</script>
```

Entre las etiquetas <body> y </body> añadimos nuestro codigo HTML

```
<body onload="init();">

<div class="view" id="view">
<div class="axis" style="visibility:hidden;"> X Axis </div>
<div class="axis" style="visibility:hidden;-webkit-transform: rotate(90deg);">
Y Axis </div>
<div class="axis" style="visibility:hidden;-webkit-transform: rotateY(-90deg);">
Z Axis </div>
<div class="face"></div>
<div class="face_principal" style="-webkit-transform: translateZ(600px)
translateY(-200px) translateX(-40px);">
<div id="cont_carcasa">
        <div id="cuerpo">
                <div id="botones">
                        <div id="boton0" class="deg">0</div>
                        <div id="boton1" class="deg">1</div>
                        <div id="boton2" class="deg">2</div>
                        <div id="boton3" class="deg">3</div>
                        <div id="boton4" class="deg">4</div>
                        <div id="boton5" class="deg">5</div>
                        <div id="boton6" class="deg">6</div>
                        <div id="boton7" class="deg">7</div>
```

```
                        <div id="boton8" class="deg">8</div>
                        <div id="boton9" class="deg">9</div>
                        <div id="boton10" class="deg">Rx</div>
                        <div id="boton11" class="deg">Tx</div>
                        <div id="boton12" class="deg">Ch</div>
                        </div>
                        <div id="boton13" class="deg"
                        onclick="visualiza(option_selected)">
                        Enter</div>
                        <div id="cara_frontal">

    <!-------------------------BOTON DISTRESS---------------------------------------->
    <div id="distress"><img src="img/boton.jpg" height="71" width="87"></div>
    <div id="boton_rojo"  class="deg2" onclick="distress()">DISTRESS</div>
    <div id="tapa" onMouseOver="levanta()" ></div>
    <!-------------------------rrallas----------------------------------------->

                        <div id="rallas">
                        <div id="ralla1"></div>
                        <div id="ralla2"></div>
                        <div id="ralla3"></div>
                        <div id="ralla4"></div>
                        <div id="ralla5"></div>
                        <div id="ralla6"></div>
                        <div id="ralla7"></div>
                        <div id="ralla8"></div>
                        <div id="ralla9"></div>
                        <div id="ralla10"></div>
                        <div id="ralla11"></div>

</div>
<div id="pantalla">

<div id="cont_pantallas">

<div id='pantalla_0' class="pantalla" cancel="" arrows=""
style="visibility:hidden"><img src="img/pantalla0.jpg" id="mar" width="350"
height="200" ></img></div>

<div id='pantalla_welcome' class="pantalla" cancel="status" arrows=""
style="visibility:hidden">
```

```
<img src="img/tierra3.jpg" id="mar" width="350" height="200" >
<div id="capa" class="welcome" ></div>
</div>

<div id='pantalla_dim'  class="pantalla" cancel="status" arrows="brillo"
style="visibility:hidden;">
 <table id="tabla1"  width="350" border="0" cellspacing="10" cellpadding="0"
bordercolor="cccccc">

<tr>
 <td colspan=4></td>

</tr>

<tr>
<td colspan=3 style="font-size:28px;">  brigtness<b>:</b></td>
<td id="texto" colspan=2 style="font-size:33px;color:red;font-
family:tahoma"><b>10</b></td>
<td></td>
<tr>
<td>  &nbspCancel</td>
<td><img src="img/flecha_abajo.png" id="mar" width="35" height="35" ></td>
<td><img src="img/flecha_arriba.png" id="mar" width="35" height="35" ></td>
<td>Save</td>
</tr>
</table>
</div>

<div id='pantalla_status' class="pantalla" cancel="status" arrows=""
style="visibility:hidden">
 <table id="tabla2" width="350" border="0" cellspacing="0" cellpadding="0"
bordercolor="cccccc">
<tr>
<!---------- reloj ---------->

<td>  MSSII<b>:  345678888    
          </b>UTC</t
d>
<td> <div id="reloj" class='reloj'
style="visibility:hidden;left:110px;top:5px;"></div></td>
</tr>
```

```
<td> </td>
<tr>
<td >  Lat<b>:  45<sup>o</sup>05<sup>'</sup>
</b>N,    long<b>:  015<sup>o</sup>17
<sup>'</sup></b>E</td>
<td> </td>
</tr>
<tr>
<td colspan=2 style="font-
size:25px;">;Rx   <b>2181.4</b>   kHz</td>
<td></td>
</tr>
<tr>
<td colspan=2 style="font-size:25px;"
>  Tx   <b>2182</b>.0   kHz</td
>
<td></td>
</tr>
<tr>

<!--- BARRA NIVEL -->

<td  style="font-size:15px;">
<div id="cont_barra"
style="visibility:hidden;width:100px;height:10px;left:135px;top:160px;border:1px
solid orange;">
<div id='barra'
style="background:orange;height:10px;visibility:hidden;"></div></div>  
;signal</td>
<td></td>
</tr>
<tr>
<td >  </td>
<td></td>
</tr>
</table>

</div>

<div id='pantalla_menu' class="pantalla" cancel="status" arrows="select_menu"
default="1" max="5" name="" style="visibility:hidden">
```

```
<table id="tabla3" width="324" border="1" cellspacing="10" cellpadding="0"
bordercolor="cccccc">
<tr id="fila1" class="fila selected"><td  align=left colspan=5>1.DSC CALL</td>
</tr>
<tr id="fila2" class="fila"><td  align=left colspan=5 >2.DSC LOG</td>  </tr>
<tr id="fila3" class="fila"><td  align=left colspan=5 >3.COMPOSED DSC
CALL</td>  </tr>
<tr id="fila4" class="fila"><td align=left colspan=5  >4.STATIONS</td></tr>
<tr id="fila5" class="fila"><td  align=left colspan=5 >5.SET UP</td>  </tr>
<tr><td><p3>Cancel</td><td><img src="img/flecha_arriba.png" id="mar"
width="35" height="35" ></img></td><td>
<img src="img/flecha_abajo.png" id="mar" width="35" height="35"
></img></td><td><p3><p3>Save</p3></td></tr>
</table>
</div>

<div id='pantalla_1' class="pantalla" cancel="menu" arrows="select_menu"
default="1_1" max="7" name="1_" style="visibility:hidden">
 <table id="tabla4"  border="0" cellspacing="0" cellpadding="0"
bordercolor="cccccc">
 <tr id="fila1_1" class="fila selected"><td  align=left colspan=5 >1.COAST
STATION</td>  </tr>
 <tr id="fila1_2" class="fila"><td  align=left colspan=5 >2.SHIP</td>  </tr>
 <tr id="fila1_3" class="fila"><td  align=left colspan=5 >3.AREA</td>  </tr>
 <tr id="fila1_4" class="fila"><td align=left colspan=5  >4.DISTRESS</td></tr>
<tr id="fila1_5" class="fila"><td  align=left colspan=5 >5.INDIVIDUAL</td>
</tr>
 <tr id="fila1_6" class="fila"><td  align=left colspan=5 >6.GROUP</td>  </tr>
<tr id="fila1_7" class="fila"><td  align=left colspan=5 >7.TEST CALL</td>
</tr>
 <tr><td><p3>Cancel</td><td><img src="img/flecha_arriba.png" id="mar"
width="35" height="35" ></img></td><td>
<img src="img/flecha_abajo.png" id="mar" width="35" height="35"
></img></td><td><p3><p3>Save</p3></td></tr>
 </table>
</div>

<div id='pantalla_2' class="pantalla" cancel="menu" arrows="select_menu"
default="1" max="2" name="2_" style="visibility:hidden">
 <table id="tabla5" width="324" border="1" cellspacing="10" cellpadding="0"
bordercolor="cccccc">
```

```
 <tr id="fila2_1" class="selected"><td  align=left colspan=5 >1.XXXX</td>
</tr>
 <tr id="fila2_2"><td  align=left colspan=5 >2.YYYYY</td>  </tr>
 <tr><td><p3>Cancel</td><td><img src="img/flecha_arriba.png" id="mar"
width="35" height="35" ></img></td><td>
<img src="img/flecha_abajo.png" id="mar" width="35" height="35"
></img></td><td><p3><p3>Save</p3></td></tr>
 </table>
</div>

<div id='pantalla_3' class="pantalla" style="visibility:hidden">3</div>
<div id='pantalla_4' class="pantalla" style="visibility:hidden">4</div>
<div id='pantalla_5' class="pantalla" style="visibility:hidden">
<img src="img/tierra2.jpg" id="mar" width="350" height="200" ></img></div>

<div id='pantalla_1_1' class="pantalla" cancel="1" arrows="select_menu"
default="1" max="2" name="2_" style="visibility:hidden">
pantalla 1_1
</div>


<!----------------------------_LAS DEMAS
PANTALLAS---------------------------------------------------->

<div id='pantalla_1_4' class="pantalla" cancel="1" arrows="select_menu"
default="1_4_1" max="2" name="1_4_" style="visibility:hidden;color:white">
<table id='6' width="324" border="1" cellspacing="10" cellpadding="0"
bordercolor="cccccc">
 <tbody>
 <tr id="fila1_4_1" class="selected" style="font-size:10px;"><td  align=left
colspan=5 >ALERT</td>  </tr>
 <tr id="fila1_4_2"><td  align=left colspan=5 >RELAY</td>  </tr>
<tr><td><p3>Cancel</td><td><img src="img/flecha_arriba.png" id="mar"
width="35" height="35" ></img></td><td>
<img src="img/flecha_abajo.png" id="mar" width="35" height="35"
></img></td><td><p3><p3>Save</p3></td></tr>
 </table>
</div>


<!--------------------------MENU 4------------------------------------>
```

```
<div id='pantalla_1_4_1' class="pantalla" cancel="1_4" arrows="select_menu"
max="3" default="1_4_1_1" name="1_4_1_"
style="visibility:hidden;color:white">

<table id="tabla6" width="324" border="1" cellspacing="10" cellpadding="0"
bordercolor="cccccc">
<tbody>
<tr id="fila1_4_1_1"><td  align=left colspan=5
><span>TYPE</span><b>:</b>DISTRESS ALERT</td>  </tr>
<tr id="fila1_4_1_2"><td  align=left colspan=5 ><span>TO</span><b>:</b>
ALL SHIPS</td></tr>
<tr id="fila1_4_1_3"><td align=left colspan=5
><span>NAT</span><b>:</b>cause of distress</td></tr>
<tr><td><p3>Cancel</td><td><img src="img/flecha_arriba.png" id="mar"
width="35" height="35" ></td>
<td><img src="img/flecha_abajo.png" id="mar" width="35" height="35" ></td>
<td><p3><p3>Save</p3></td></tr>
</table>
</div>

<!---------------------------MENU NATURE--------------------------------------->
<div id='pantalla_1_4_1_3' class="pantalla" cancel="1_4_1"
arrows="select_menu" max="7" default="1_4_1_3_1" name="1_4_1_3_"
style="visibility:hidden;color:white">
<table id="tabla7"  border="0" cellspacing="10" cellpadding="0"
bordercolor="cccccc">

<tr id="fila1_4_1_3_1" class="selected" style="font-size:10px;"><td  align=left
colspan=5 >FIRE</td>  </tr>
<tr id="fila1_4_1_3_2"><td align=left colspan=5 >EXPLOSION</td>  </tr>
<tr id="fila1_4_1_3_3"><td align=left colspan=5 >CAPSIZING</td></tr>
<tr id="fila1_4_1_3_4"><td align=left colspan=5 >PIRACY ATACK</td></tr>
<tr id="fila1_4_1_3_5"><td align=left colspan=5 >AGROUND</td></tr>
<tr id="fila1_4_1_3_6"><td align=left colspan=5 >FLOWING</td></tr>
<tr id="fila1_4_1_3_7"><td align=left colspan=5 >LISTING</td></tr>
<tr><td><p3>Cancel</td><td><img src="img/flecha_arriba.png" id="mar"
width="35" height="35" ></td>
<td><img src="img/flecha_abajo.png" id="mar" width="35" height="35" ></td>
<td><p3><p3>Save</p3></td></tr>
</table>
```

```
</div>

<!--------------------------info--------------------------------------->

<div id='pantalla_info' class="pantalla" cancel="1_4_1_3" arrows="select_menu"
max="7" default="call" name="info_" style="visibility:hidden;color:white">
<table id="tabla_info" width="324" border="1" cellspacing="10" cellpadding="0"
bordercolor="cccccc">
<tr id="fila7_0"><td  align=left colspan=5 style="font-size:10px;"></td></tr>
<tr id="fila7_1"><td  align=left
colspan=5><span>TYPE</span><b>:</b>DISTRESS ALERT</td></tr>
<tr id="fila7_2"><td  align=left colspan=5><span>TO</span><b>: </b>ALL
SHIPS</td></tr>
<tr id="fila7_3"><td  align=left
colspan=5><span>NAT</span><b>:</b>ABANDONING SHIP</td></tr>
<tr id="fila7_4"><td  align=left colspan=5><span>MODE</span><b>:</b>SSB
TELEPHONY</td></tr>
<tr id="fila7_2"><td  align=left colspan=5><span>DSC</span><b>:</b>Tx/Rx
<b>2187.5</b></td></tr>
<tr id="fila7_3"><td  align=left
colspan=5><span>LAT</span><b>:45<sup>o</sup>05<sup>'</sup></td></tr>
<tr id="fila7_4"><td  align=left
colspan=5><span>LONG</span><b>:015<sup>o</sup>17<sup>'</sup>E</b>
</td><          /tr>
<tr id="fila7_4"><td  align=left
colspan=5><span>UTC</span><b>:09:26</td></tr>
<tr><td><p3>Cancel</td>
</table>
</div>


<!--------------------------MENU CALL------------------------------->

<div id='pantalla_call'  class="pantalla" cancel="info" arrows="select_menu"
max="7" default="" name="call_"  style="visibility:hidden;color:white">
<table id="table_call" width="324" border="1" cellspacing="10" cellpadding="0"
bordercolor="cccccc" style="background-color:#000">
<tr id="fila8_0"><td  align=left colspan=5 style="font-size:10px;"></tr>
<tr id="fila8_1"><td  align=left colspan=5></td></tr>
<tr id="fila8_2"><td  align=left colspan=5></td></tr>
<tr id="fila8_3"><td  align=left colspan=5><span></span><b></b></td></tr>
```

```
<tr id="fila8_4"><td  align=left colspan=5><div id="capa22" class="mensaje"
style="left:10px;top:5px;visibility:hidden" >
</div></td></tr>
<tr id="fila8_2"><td  align=left colspan=5></td></tr>
<tr id="fila8_3"><td  align=left colspan=5></td></tr>
<tr id="fila8_4"><td  align=left colspan=5></tr>
<tr id="fila8_4"><td  align=left colspan=5>cancel</tr>
</table>
</div>


<!--------------------------MENU distress------------------------------>

<div id='pantalla_distress'  class="pantalla" cancel="info" arrows="select_menu"
max="7" default=" " name="distress_"  style="visibility:hidden;color:white">
<table id="table_distress" width="324" border="1" cellspacing="10"
cellpadding="0" bordercolor="cccccc" style="background-color:#000">
<tr id="fila8_0"><td  align=left colspan=5 style="font-size:10px;"></tr>
<tr id="fila8_1"><td  align=left colspan=5></td></tr>
<tr id="fila8_2"><td  align=left colspan=5></td></tr>
<tr id="fila8_3"><td  id="distress2"  align=left colspan=5 style="font-
size:28px"><span>DISTRESS</span><b></b></td></tr>
<tr id="fila8_4"><td  align=left colspan=5></tr>
<tr id="fila8_2"><td  align=left colspan=5></td></tr>
<tr id="fila8_3"><td  align=left colspan=5></td></tr>
<tr id="fila8_4"><td  align=left colspan=5></tr>
<tr id="fila8_4"><td  align=left colspan=5>cancel</tr>
</table>
</div>



</div></div>


<div id="botones_horizontales">
<div id="boton0" class="deg" onclick="cancel();">.</div>
<div id="boton1" class="deg"  onclick="setvalue('down')">.</div>
<div id="boton2"  style="z-index:34;" class="deg"
onclick="setvalue('up')">.</div>
<div id="boton3" class="deg" onclick="visualiza('status')">.</div>
```

```
</div>

<div id="botones_verticales" >
<div id="boton4"  class="deg" onclick=visualiza('dim')>Dim</div>
<div id="boton5" class="deg">Mode</div>
<div id="boton6" class="deg"
onclick="clearTimeout(bucle_welcome);visualiza('menu')">Menu</div>

<div id="power"  class="deg" onclick="power()">Power</div>

</div>

</div></div></div>


</div>
<div class="esquinas"
 style="-webkit-transform: translateZ(300px) translateX(450px)
rotateY(90deg);">
 </div>
<div class="esquinas"
  style="-webkit-transform: translateZ(300px) translateX(-300px) rotateY(-
90deg);">
  </div>
<div class="face2"
  style="-webkit-transform: translateY(30px)  translateX(-80px) translateZ(300px)
rotateX(90deg);">
  </div>
<div class="face" id="top" style="-webkit-transform: translateY(-600px) rotateX(-
90deg);">
  <p>Sailor</p></div>

<div  class="circuito" id="circuito" style="transform:translateY(30px)
translateX(-80px) translateZ(300px) rotateX(90deg);-webkit-transform:
translateY(30px)  translateX(-80px) translateZ(300px) rotateX(90deg);">
  <img src="img/9.jpg" height="600" width="750"/></div>

</div>

<div id="contenedor">
<ul id="menu">
```

```
<li></li>
 <li>
        <a href="#"><p3>Controles</p3></a>
 <ul class="children">
<div id="controles"><div id="cont"><br>RotateX:<input id="xSlide"
type="range"
          min="-180" max="180" step="5" value="-10" onchange="setXY()">
                <br>RotateY:<input id="ySlide" type="range"
          min="-180" max="180" step="5" value="20" onchange="setXY()">
<br>RotateZ:<input id="zSlide" type="range"
          min="-180" max="180" step="5" value="0" onchange="setXY()">
<br>TX:<input id="TXSlide" type="range"
          min="-180" max="180" step="5" value="-180" onchange="setXY()">
<br>TY: <input id="TYSlide" type="range"
          min="-180" max="180" step="5" value="100" onchange="setXY()">


<p><input type="button"  class="deg" value="Open" onclick="openBox();">
<input type="button"   class="deg" value="Close" onclick="closeBox();">
<br>
<input type="button"   class="deg" value="togle" onclick="togleAxis();">
<input type="button"   class="deg" value="tapa" onclick="baja_tapa();">

</div>
</ul>
</div>


<div id="capa2" style='visibility:hidden'></div>
```

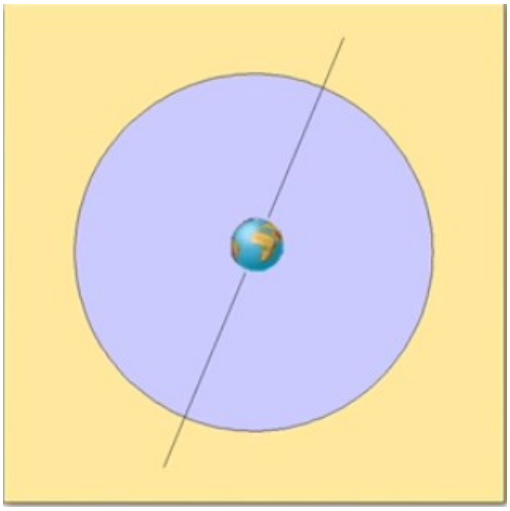
Finalmente añadimos las etiquetas:

```
</body>
</html>
```

Elementos de la esfera celeste con javascript

Recordemos algunas de la definiciones de los elementos de la esfera celeste clasificados en la tabla siguiente por elementos fijos,es decir que no dependen del observador o los que si dependen de el observador o de una posición dada para cada lugar de la tierra.

Consideremos a la tierra en el centro de una gran esfera celeste como muestra la siguiente imagen:



la tierra en el centro de la esfera celeste

El eje del mundo: es la prolongación del eje de rotación terrestre que corta a la esfera celeste en dos puntos: Polo norte celeste y polo sur celeste.

El plano del ecuador: es perpendicular al eje del mundo,pasa por el centro de la esfera y la divide en dos hemisferios, norte o Boreal, sur o Austral.

Meridianos celestes: son planos o círculos máximos perpendiculares al plano del ecuador que contienen al eje del mundo. Existen infinitos meridianos en una esfera.

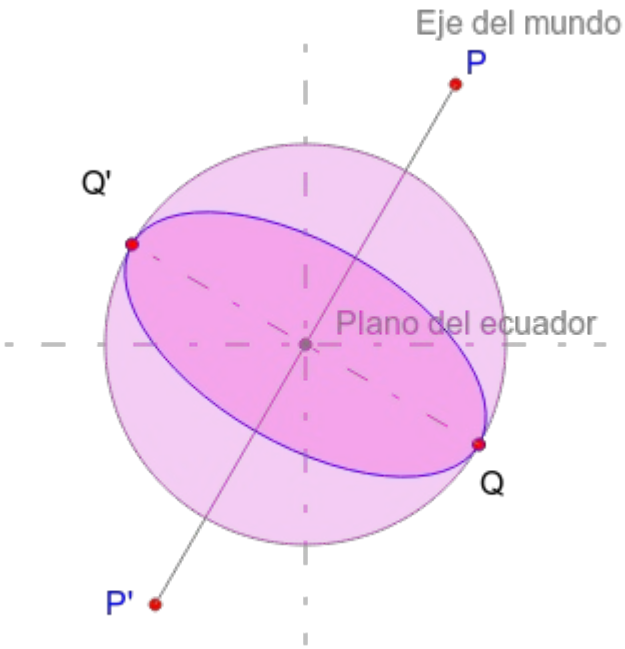
Pasemos ahora a describir los elementos de la esfera que son peculiares a una situación o posición de un observador:

Linea Zenit-Nadir o vertical del lugar: linea perpendicular a la superficie de la esfera,que pasa por su centro y es la que marcaría una plomada,corta a la esfera en dos puntos que dependen de la posición del observador *Zenit* y *Nadir.*

Plano del Horizonte:es perpendicular a la vertical del lugar o la linea zenit nadir y divide a la esfera en dos hemisferios, el visible y el no visible, una estrella sera visible si esta por encima del horizonte o no visible si se encuentra por debajo de este.

El primer script : el eje del mundo y el plano del ecuador

Vamos a dibujar pues, el eje del mundo y el plano del ecuador perpendicular a este, con código javascript:



El primer Script: El plano del Ecuador perpendicular al eje del mundo

El código es el siguiente:

```
<html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Esfera Celeste</title>
<script src="basic.js" type="text/javascript"></script>
<style>
canvas{
position: absolute;
top: 0px;
left:0px;
border: 1px solid gray;
}
</style>
</head>
<body onload='init()'>
<canvas id="canvas" width="400" height="400"></canvas>
<script>
var canvas,
ctx,rad,centro,inclinacion;
var q,q2,Q,Q2,radio;

function init() {
canvas = document.getElementById("canvas");
ctx = canvas.getContext('2d');
ctx.strokeStyle = 'purple';
ctx.lineWidth = 6;
ctx.lineCap = 'round';
rad=Math.PI/180;
centro={x:canvas.width/2,y:canvas.height/2};
radio={x:(canvas.height/2)-100,y:((canvas.height/2)-100)/2};;
radio2={x:(canvas.height/2)-50,y:(canvas.height/2)-50};
inclinacion=30;//Ecuador

draw();
//Point(ctx,pos,radio,grosor,color1,color2)
//DashedLine(ctx,objeto1,objeto2,grosor,color1)
}
```

```
function draw(){
new
Circulo(ctx,centro.x,centro.y,radio.x,radio.x,0,360,1,'grey','rgba(200,0,200,.2)');
//ESFERA grande
Centro=new Point(ctx,centro,3,1,'grey','grey');
ecuador(inclinacion);
Ejes(centro,(canvas.height/2)-50)
}
</script>
</body>
</html>
```

Encontramos la misma estructura de los códigos anteriores, una etiqueta

<html>

<head>

<title></title>

<script src="basic.js" type="text/javascript"></script> esta es la ruta donde tenemos el fichero externo **basic.js**

<style> es la declaración de estilos que aplicaremos.

<body onload='init()'> al cargarse el documento la primera función que ejecutara el navegador serà **'init()'** y esta llamara a la función **draw()** después de haber sido declaradas las variables en el bloque general del encabezamiento de programa.

<script></script> Escribimos el código entre estas etiquetas.

Podemos hacer comentarios aclaratorios en cualquier punto del programa escribiendo primero // el navegador se saltara todo el texto después de la dos barras pues no lo considerará código ejecutable.

Veamos ahora el código **basic.js** en el que están todas las funciones que necesitamos para la ejecución de nuestro programa:

```
function ecuador(angle){

var PoloN=formula(centro,radio2,inclinacion+270);
var PoloS=formula(centro,radio2,inclinacion+90);
var poloN=new Point(ctx,PoloN,3,1,'grey','red');
var poloS=new Point(ctx,PoloS,3,1,'grey','red');
Q=formula2(centro,radio,0,(angle)*rad);
Q2=formula2(centro,radio,180,(angle)*rad);
ElipseInclinada(centro,radio.x,radio.y,angle,'blue','rgba(250,0,200,.2)');//Ecuador
line(ctx,poloN,poloS,1,'grey');
q=new Point(ctx,Q,3,1,'grey','red');
q2=new Point(ctx,Q2,3,1,'grey','red');
Enumera([{x:poloN.x+5,y:poloN.y-5},{x:poloS.x-25,y:poloS.y+5},
{x:q.x,y:q.y+25},{x:q2.x-25,y:q2.y-25}],
['P',"P'",'Q',"Q'"],['blue','blue','black','black'],[16,16,16,16]);
DashedLine(ctx,q,q2,1,'grey'); // linea que une q con q'
}
function horizonte(theta,num){

cardinales=[];
var Zenit=formula(centro,radio2,inclinacion2+270);
var Nadir=formula(centro,radio2,inclinacion2+90);
//var zenit=new Point(ctx,Zenit,3,1,'grey','blue');
//Point(ctx,pos,radio,grosor,color1,color2)
//var nadir=new Point(ctx,Nadir,3,1,'grey','blue');
//Point(ctx,pos,radio,grosor,color1,color2)

for(var i=0;i<360;i+=num){
 cardinales.push(new
Point(ctx,formula2(centro,radio,i+10,theta*rad),3,1,'grey','red'));

}

for(var i=0;i<cardinales.length-1;i++){
//line(ctx,cardinales[i],cardinales[i+1],'grey');//Linea Perimetro

}
//line(ctx,cardinales[0],cardinales[cardinales.length-1],'grey');//Linea Perimetro

ElipseInclinada(centro,radio.x,radio.y,inclinacion2,'blue','rgba(20,100,250,.2)');
//Horizonte
```

```
line(ctx,Zenit,Nadir,.5,'blue');//linea Z-N
DashedLine(ctx,cardinales[0],cardinales[2],1,'grey');//linea N-S
DashedLine(ctx,cardinales[3],cardinales[1],1,'grey');//Linea Meridiana E-W

Enumera([{x:cardinales[0].x+10,y:cardinales[0].y},//E
{x:cardinales[1].x,y:cardinales[1].y+18},   //S
{x:cardinales[2].x-20,y:cardinales[2].y},//W
{x:cardinales[3].x,y:cardinales[3].y-15},//N
Zenit,Nadir
],
['E','S','W','N','Zenit','Nadir'],['blue','blue','blue','blue','navy','navy'],
[10,10,10,10,12,12]);
}
function EjeHorizontal(centro,long){
DashedLine(ctx,{x:centro.x-long,y:centro.y},
{x:centro.x+long,y:centro.y},1,'grey');
//Eje Horizontal
}
function EjeVertical(centro,long){
DashedLine(ctx,{x:centro.x,y:centro.y-long},
{x:centro.x,y:centro.y+long},1,'grey');
//Eje Vertical
}

function Enumera(array1,array2,color,size){
for(var i=0;i<array2.length;i++){
escribe2(ctx,array2[i],{x:array1[i].x,y:array1[i].y},color[i],size[i])
}
escribe2(ctx,'Plano del ecuador',{x:centro.x+15,y:centro.y-5},'grey',16)
escribe2(ctx,'Eje del mundo',{x:centro.x+55,y:centro.y-155},'grey',16)
}

function escribe2(ctx,mensaje,pos,color2,tam){
ctx.save();
ctx.lineWidth=1;
//ctx.strokeStyle=color1;
ctx.fillStyle=color2;
ctx.font = tam+"px Arial";
ctx.fillText(mensaje,pos.x,pos.y);
//ctx.strokeText(mensaje,pos.x,pos.y);
```

```
ctx.restore();
}
function formula(pos,r,angle){
var res={x:pos.x+Math.cos(angle*rad)*r.x,y:pos.y+Math.sin(angle*rad)*r.y}
return res;
}

function formula2(pos,r,a,theta){

var x = pos.x + r.x*Math.cos(a*rad)*Math.cos(theta) -
r.y*Math.sin(a*rad)*Math.sin(theta);
var y = pos.y + r.x*Math.cos(a*rad)*Math.sin(theta) +
r.y*Math.sin(a*rad)*Math.cos(theta);
return {x:x,y:y};

}

function ElipseInclinada(c,rx,ry,theta,color,color2){

theta = theta*rad;
ctx.strokeStyle=color;
ctx.fillStyle=color2;
ctx.beginPath();
for (var a = 0; a < 360; a+=6) {
var x = c.x + rx*Math.cos(a*rad)*Math.cos(theta) -
ry*Math.sin(a*rad)*Math.sin(theta);
var y = c.y + rx*Math.cos(a*rad)*Math.sin(theta) +
ry*Math.sin(a*rad)*Math.cos(theta);
ctx.lineTo(x, y);
}
ctx.closePath();
ctx.stroke();
ctx.fill();
}

function Point(ctx,pos,radio,grosor,color1,color2){
this.ctx=ctx;
this.x=pos.x;
this.y=pos.y;
this.radio=radio;
this.grosor=grosor
```

```
this.color=color1;
this.color2=color2;
this.dibuja();
}
Point.prototype.dibuja=function(){

this.ctx.save();
this.ctx.lineWidth=this.grosor;
this.ctx.strokeStyle=this.color1;
this.ctx.fillStyle=this.color2;
this.ctx.beginPath();
this.ctx.arc(this.x,this.y,this.radio,0,2*Math.PI,false);
this.ctx.stroke()
this.ctx.fill();
this.ctx.restore();
}


function Circulo(ctx,x,y,radioX,radioY,inicio,fin,grosor,color,color2){
this.ctx=ctx;
this.x=x;
this.y=y;
this.radioX=radioX;
this.radioY=radioY;
this.inicio=inicio;
this.fin=fin;
this.grosor=grosor
this.color=color;
this.color2=color2;
this.dibuja();
}
Circulo.prototype.dibuja=function(){
var radianes=Math.PI/180;
this.ctx.lineWidth=this.grosor;
this.ctx.strokeStyle=this.color;
this.ctx.fillStyle=this.color2;
this.ctx.beginPath();

for (var i=this.inicio;i<this.fin+1;i++)
{
```

```
xx =this.x +this.radioX*Math.cos(i*radianes);
yy = this.y +this.radioY*Math.sin(i*radianes);
this.ctx.lineTo(xx,yy);

}
this.ctx.stroke()
this.ctx.fill();

}

function DashedCirculo(ctx,x,y,radioX,radioY,inicio,fin,grosor,color,color2){
this.ctx=ctx;
this.x=x;
this.y=y;
this.radioX=radioX;
this.radioY=radioY;
this.inicio=inicio;
this.fin=fin;
this.grosor=grosor
this.color=color;
this.color2=color2;
this.dibuja();
}

DashedCirculo.prototype.dibuja=function(){

var radianes=Math.PI/180;
this.ctx.lineWidth=this.grosor;
this.ctx.strokeStyle=this.color;
this.ctx.fillStyle=this.color2;
//this.ctx.setLineDash([4, 14]);
//this.ctx.setLineDash([4, 14,18]);
this.ctx.setLineDash([4, 14, 12, 16]);
this.ctx.beginPath();

for (var i=this.inicio;i<this.fin+1;i++)
{

xx =this.x +this.radioX*Math.cos(i*radianes);
yy = this.y +this.radioY*Math.sin(i*radianes);
this.ctx.lineTo(xx,yy);
```

```
}

this.ctx.stroke()
this.ctx.fill();

}

function DashedLine(ctx,objeto1,objeto2,grosor,color1){
ctx.save();
ctx.strokeStyle=color1;
 ctx.lineWidth=grosor;
 ctx.beginPath();
ctx.setLineDash([4, 14, 12, 16]);
ctx.moveTo(objeto1.x,objeto1.y);
 ctx.lineTo(objeto2.x,objeto2.y);
ctx.stroke();
ctx.restore();
}

function line(ctx,objeto1,objeto2,grosor,color1){
ctx.save();
ctx.strokeStyle=color1;
 ctx.lineWidth=grosor;
 ctx.beginPath();
 ctx.moveTo(objeto1.x,objeto1.y);
 ctx.lineTo(objeto2.x,objeto2.y);
ctx.stroke();
ctx.restore();
}
```

El objeto Point(ctx,pos,radio,grosor,color1,color2) como su nombre indica dibuja un punto,recibe 6 argumentos:

ctx:es el contexto del canvas o entorno gráfico de javascript que reconoce el navegador para dibujar gráficos, note que lo hemos declarado anteriormente como variable en el encabezamiento del programa entre las etiquetas <script></script>

pos:posición donde vamos a dibujar el punto.

radio:radio del punto.

color1:color del contorno del punto.

color2:color de relleno del punto.

La función line (ctx,objeto1,objeto2,grosor,color1) recibe como parámetros:

ctx:contexto del canvas.

Objeto1:objeto que le pasa las coordenadas cartesianas de origen de la recta ejemplo {x:200,y:100}

Objeto2:objeto que le pasa las coordenadas cartesianas del fin de la recta ejemplo {x:300,y:100}

grosor,color1:definirán el estilo gráfico de nuestra recta.

De igual manera con DashedLine(ctx,objeto1,objeto2,grosor,color1) dashed en ingles significa discontinuo es una linea de trazo discontinuo que utilizamos en nuestro gráfico.

La función formula2(pos,r,a,theta) recibe los siguientes parámetros:

pos:posición.

r:radio.

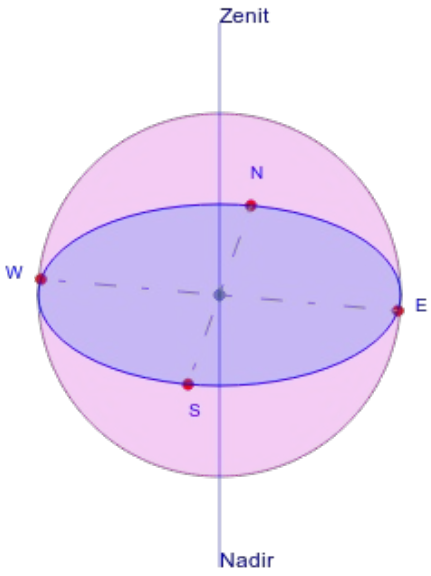
a:angulo.

theta:angulo de inclinación

Esta función lo que hace es básicamente que recibe un objeto ,posición en coordenadas polares (radio,angulo) y las transforma en cartesianas,es decir las parametriza con respecto a **a** que en este caso dibujara las **elipses** que necesitamos.

A si pues el archivo externo **basic.js** tiene todas las funciones que he creído convenientes para dibujar los elementos de la esfera celeste y las he 'diseñado' para facilitar el trabajo y no repetir lineas de código,usted con la practica y experiencia desarrollará las suyas propias que le parezcan mas aclaratorias y mas útiles para sus propósitos.

Segundo script: la linea zenit-nadir y el horizonte como plano perpendicular a ella.



Tiene básicamente el mismo código que el anterior script ,pero este en la funcion **draw()** llamara a la función **horizonte(theta,num)** que recibe dos parametros:

theta:angulo de inclinación del plano del horizonte.

num:el numero con el que subdividiremos el plano del horizonte en este caso ,una elipse con cuatro subdivisiones (cada 90 grados) ,que serán los cuatro puntos cardinales : N,S,E,W.

```
<script>
var canvas,
ctx,rad,centro,inclinacion,inclinacion2;
var Zenit,Nadir,zenit,nadir,q,q2,Q,Q2,radio;
function init() {
canvas = document.getElementById("canvas");
ctx = canvas.getContext('2d');
ctx.strokeStyle = 'purple';
ctx.lineWidth = 6;
ctx.lineCap = 'round';
rad=Math.PI/180;
centro={x:canvas.width/2,y:canvas.height/2};
radio={x:(canvas.height/2)-100,y:((canvas.height/2)-100)/2};;
radio2={x:(canvas.height/2)-50,y:(canvas.height/2)-50};
inclinacion=30;//Ecuador
inclinacion2=0;//Horizonte
draw();
//Cardinales(inclinacion2,90);
}
function draw(){
new
Circulo(ctx,centro.x,centro.y,radio.x,radio.x,0,360,1,'grey','rgba(200,0,200,.2)');
//ESFERA grande
var Centro=new Point(ctx,centro,3,1,'grey','grey');
//Point(ctx,pos,radio,grosor,color1,color2)
horizonte(inclinacion2,90);
}
</script>
```

llamara pues a la función **horizonte(theta,num)** pasandole como **theta**: 0 grados y **num**: 90 grados recuerde que la funcion horizonte() se encuentra en el archivo externo **basic.js**, declarado en el encabezamiento del script.

```
function horizonte(theta,num){
cardinales=[];
```

```
        var Zenit=formula(centro,radio2,inclinacion2+270);
        var Nadir=formula(centro,radio2,inclinacion2+90);
        //var zenit=new Point(ctx,Zenit,3,1,'grey','blue');
//Point(ctx,pos,radio,grosor,color1,color2)
        //var nadir=new Point(ctx,Nadir,3,1,'grey','blue');
//Point(ctx,pos,radio,grosor,color1,color2)
        for(var i=0;i<360;i+=num){
         cardinales.push(new
Point(ctx,formula2(centro,radio,i+10,theta*rad),3,1,'grey','red'));
        }

        for(var i=0;i<cardinales.length-1;i++){
        //line(ctx,cardinales[i],cardinales[i+1],'grey');//Linea Perimetro

        }
        //line(ctx,cardinales[0],cardinales[cardinales.length-1],'grey');//Linea Perimetro


ElipseInclinada(centro,radio.x,radio.y,inclinacion2,'blue','rgba(20,100,250,.2)');//Horiz
onte

        line(ctx,Zenit,Nadir,.5,'blue');//linea Z-N
        DashedLine(ctx,cardinales[0],cardinales[2],1,'grey');//linea N-S
        DashedLine(ctx,cardinales[3],cardinales[1],1,'grey');//Linea Meridiana E-W

        Enumera([{x:cardinales[0].x+10,y:cardinales[0].y},//E
        {x:cardinales[1].x,y:cardinales[1].y+18},   //S
        {x:cardinales[2].x-20,y:cardinales[2].y},//W
        {x:cardinales[3].x,y:cardinales[3].y-15},//N
        Zenit,Nadir
        ],
        ['E','S','W','N','Zenit','Nadir'],['blue','blue','blue','blue','navy','navy'],
        [10,10,10,10,12,12]); }
```

Conclusión

La física encapsula las leyes de la naturaleza en forma matemática. Estas leyes son simples y pueden codificarse fácilmente. Por lo tanto, generalmente es fácil crear efectos que parezcan realistas. De esta manera la programación implica cuatro pasos: identificar qué principios físicos se necesitan, anotar las ecuaciones relevantes, diseñar un algoritmo numérico para resolver las ecuaciones, y escribir el código.

Así que implica conocimientos y habilidades en cuatro áreas diferentes: física, matemáticas, métodos numéricos y programación. He intentado proporcionar una rápida visión general de los temas seleccionados en JavaScript y HTML5, enfatizando aspectos que son especialmente relevantes para la programación de las matemáticas y aplicarlo a los gráficos de curvas polares y parametricas y su aplicación en el ámbito de la náutica con varios ejemplos.

Bibliografia

Grassman Algebra for Game Development (Eric Lengyel)

Random Numbers (Squirrel Eiserloh)

Working with 3D Rotations (Stan Melax)

Inverse Kinematics (Gino van den Bergen)

Spatial Subdivision (Graham Rhodes)

Introduction to Frames, Dictionaries and K-SVD (Jim Van Verth)

Dictionary Learning in Games (Manny Ko)

http://euler.ciens.ucv.ve/pijeira/decadencia.html

http://www.arrakis.es/~mcj/heron.htm

http://almez.pntic.mec.es/~agos0000/Heron.htm

Barnet, Rich, Geometría analítica, México, McGraw-Hill, 2004.

Fuller y Tarmater. Geometría analítica, México,Addison Wesley

Iberoamericana, 2000.

Fuller, Geometría Analítica, México, Cecsa, 1999.