

Representación gráfica y manipulación matemática de vectores bidimensionales en ROOT

Andrés M. Vargas¹

¹Grupo de Investigación en Física e Informática
Universidad Distrital Francisco José de Caldas

Semana de la Enseñanza de la Física, 2012

Índice

1 Introducción

- Objetivo
- ¿Qué es ROOT?

2 Programación

- ¿Qué necesitamos?
- Creación y configuración del tablero
- Ejes coordenados
- Declarar Vectores
- Representación gráfica de Vectores

3 Conclusión

Índice

1 Introducción

- Objetivo
- ¿Qué es ROOT?

2 Programación

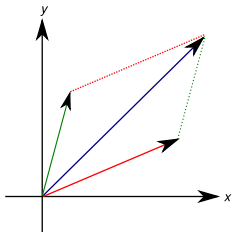
- ¿Qué necesitamos?
- Creación y configuración del tablero
- Ejes coordenados
- Declarar Vectores
- Representación gráfica de Vectores

3 Conclusión

Objetivo

❶ Objetivo General

- Presentar parte de la sección gráfica de la infraestructura de análisis de datos ROOT



❶ Objetivos Específicos

- Representar gráficamente vectores bidimensionales a través de la infraestructura ROOT, utilizando para ello las clases TCanvas, TGaxis, TVector2, TArrow y TLine.
- Comprender la estructura sintáctica de los constructores de las clases y de los métodos utilizados.



Índice

1 Introducción

- Objetivo
- ¿Qué es ROOT?

2 Programación

- ¿Qué necesitamos?
- Creación y configuración del tablero
- Ejes coordenados
- Declarar Vectores
- Representación gráfica de Vectores

3 Conclusión

ROOT

- ¿Qué es ROOT?
 - Infraestructura LGPL
 - Análisis de datos
 - Desarrollo de Software
 - CERN, 2002.
 - CINT & Clases de C++



Índice

1 Introducción

- Objetivo
- ¿Qué es ROOT?

2 Programación

- ¿Qué necesitamos?
- Creación y configuración del tablero
- Ejes coordenados
- Declarar Vectores
- Representación gráfica de Vectores

3 Conclusión

Objetos Necesarios

- Tablero (*TCanvas*)
- Sistema Coordinado (*TGaxis*)
 - Eje X
 - Eje Y
- Vectores (*TVector2*)
 - Representación gráfica (*TArrow*)
- Líneas de proyección (*TLine*)

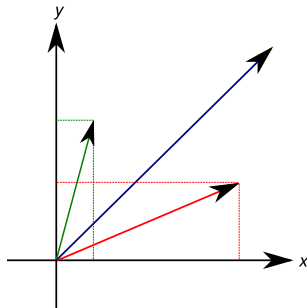
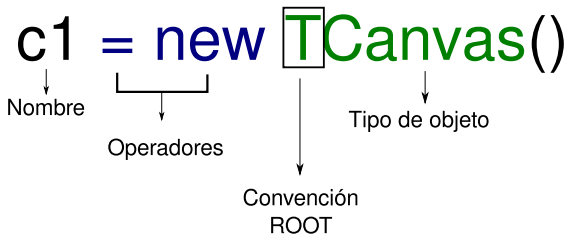


Figura: Objetos necesarios

Índice

- 1 Introducción
 - Objetivo
 - ¿Qué es ROOT?
- 2 Programación
 - ¿Qué necesitamos?
 - Creación y configuración del tablero
 - Ejes coordenados
 - Declarar Vectores
 - Representación gráfica de Vectores
- 3 Conclusión

Crear lienzo



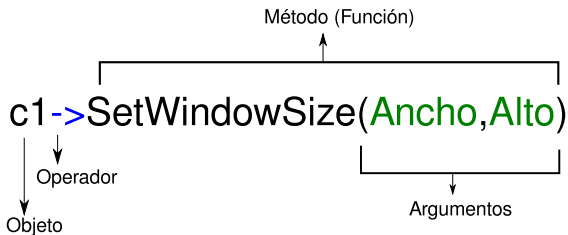
1 Ejecutar ROOT

Ejecución de ROOT

```
user@machine: $ root -l
```

- Clases, y métodos.
- El operador ->

Configurar lienzo

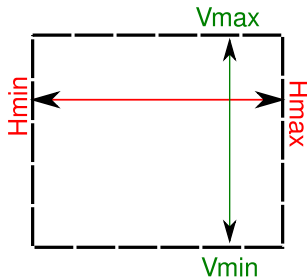


- Métodos
- Argumentos
- Tipos de argumentos

Configurar lienzo

```
c1->Range(Hmin,Vmin,Hmax,Vmax)
```

```
c1->Range(-10,-10,10,10)
```



Rango

```
c1->Range(-10,-10,10,10)
```

Índice

1 Introducción

- Objetivo
- ¿Qué es ROOT?

2 Programación

- ¿Qué necesitamos?
- Creación y configuración del tablero
- Ejes coordenados
- Declarar Vectores
- Representación gráfica de Vectores

3 Conclusión

Creación de los Ejes

- ¿Qué variables describen (caracterizan) un eje?

```
EX = new TGaxis(PInicial,PFinal,min,max)
```

```
EX = new TGaxis(-10,0,10,0,-10,10)
```

```
EY = new TGaxis(0,-10,0,10,-10,10)
```

Creación de los Ejes

```
EX = new TGaxis(-10,0,10,0,-10,10);  
EY = new TGaxis(0,-10,0,10,-10,10);  
EX->Draw();  
EY->Draw();
```



Resultado

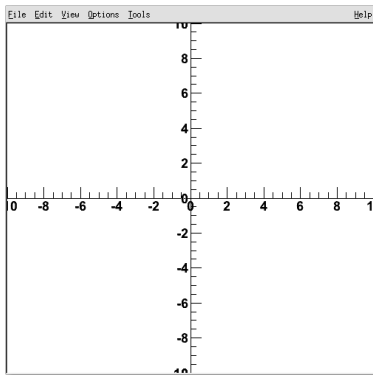


Figura: Ejes Coordinados

Índice

1 Introducción

- Objetivo
- ¿Qué es ROOT?

2 Programación

- ¿Qué necesitamos?
- Creación y configuración del tablero
- Ejes coordenados
- **Declarar Vectores**
- Representación gráfica de Vectores

3 Conclusión

$$\begin{aligned}\vec{R}_1 &= 4,3\hat{i} + 5,6\hat{j} \\ &= 4,3 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 5,6 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} 4,3 \\ 5,6 \end{bmatrix} \\ \vec{R}_2 &= -1,3\hat{i} + 4,8\hat{j} \\ &= \begin{bmatrix} -1,3 \\ 4,8 \end{bmatrix} \\ \vec{R}_3 &= \vec{R}_1 + \vec{R}_2 \\ &= \begin{bmatrix} 4,3 \\ 5,6 \end{bmatrix} + \begin{bmatrix} -1,3 \\ 4,8 \end{bmatrix} \\ &= \begin{bmatrix} -3 \\ 10,4 \end{bmatrix}\end{aligned}$$


Declarar vectores

- Vector: Un nuevo tipo de variable

TVector2 r1,r2,r3 float a,b,c

$R.\text{Set}(C_x, C_y)$ $\left\{ \begin{array}{l} r1.\text{Set}(4.3,5.6) \\ r2.\text{Set}(-1.3,4.8) \end{array} \right.$
↓ Operador

$r3 = r1 + r2$

- El método Set() de la clase TVector2.

Declarar vectores

```
TVector2 r1,r2,r3  
r1.Set(4.3,5.6)  
r2.Set(-1.3,4.8)  
r3 = r1 + r2
```

Hablando con TVector2

- Componentes del vector
 - Métodos $X()$ y $Y()$

Mostrar componentes en pantalla

```
cout << r3.X() << endl
cout << r3.Y() << endl
printf("Cx = %f \t Cy
= %f", r3.X(), r3.Y())
```

- Módulo
 - Método *Mod()* y *Mod2()*

Módulo y norma de un Vector

```
printf("|r3|= %f", r3.Mod())
```

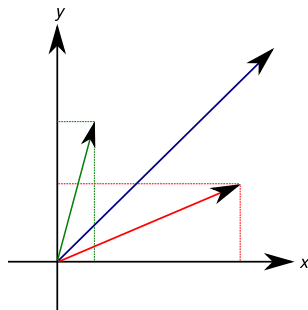


Figura: Objetos necesarios

Índice

1 Introducción

- Objetivo
- ¿Qué es ROOT?

2 Programación

- ¿Qué necesitamos?
- Creación y configuración del tablero
- Ejes coordenados
- Declarar Vectores
- Representación gráfica de Vectores

3 Conclusión

Representación Gráfica

- ¿Cómo es la representación gráfica de vectores bidimensionales?

Representación Gráfica

```
A1 = new TArrow(0,0,r1.X(),r1.Y());  
A1->SetLineColor(1);  
A1->Draw();
```

```
A1 = new TArrow(PInicial, PFinal)  
A1 = new TArrow(0,0,Cx,Cy)  
A1 = new TArrow(r1.X(),0,r1.X(),r1.Y())
```

- Ejercicio: Representar gráficamente r2 y r3

Líneas de Proyección

```
L1x = new TLine(Pinicial,Pfinal)
L1x = new TLine(r1.X(),0,r1.X(),r1.Y())
L1x->Draw()
```

Líneas de Proyección

```
L1x = new TLine(r1.X(),0,r1.X(),r2.X())
L1y = new TLine(0,r1.Y(),r1.X(),r2.X())
L1x->Draw()
L1y->Draw()
```

- Desarrollo de la interfaz gráfica del programa

Interfaz Gráfica

```
user@machine:~$root -l Escritorio/Sources/Vector.C
```



Conclusión

- ROOT es una infraestructura que permite crear prototipos de software de manera ágil, facilita la escritura de software para la enseñanza de la física al implementar clases de uso frecuente en el área.

Lecturas sugeridas



ROOT Users Guide 5.26, *The ROOT Team*. (2009)



C++ Language Tutorial, *Juan Soulié*. (2007)