**Manual de Usuario**

**Paleta de Componentes**

**LazFilters**

Contenido

[Visión General de los Componentes 1](#_Toc295672945)

[Units 2](#_Toc295672946)

[Tipos de datos 3](#_Toc295672947)

[Estructura de Clases 4](#_Toc295672948)

[Clase TFilter 4](#_Toc295672949)

[Diseño 4](#_Toc295672950)

[Propiedades 4](#_Toc295672951)

[Métodos 4](#_Toc295672952)

[Clase TFilterIIR 6](#_Toc295672953)

[Diseño 6](#_Toc295672954)

[Propiedades 6](#_Toc295672955)

[Clase TFilterIIRLP 7](#_Toc295672956)

[Diseño 7](#_Toc295672957)

[Propiedades 7](#_Toc295672958)

[Métodos 7](#_Toc295672959)

[Clase TFilterIIRHP 8](#_Toc295672960)

[Diseño 8](#_Toc295672961)

[Propiedades 8](#_Toc295672962)

[Métodos 8](#_Toc295672963)

[Clase TFilterIIRBP 9](#_Toc295672964)

[Diseño 9](#_Toc295672965)

[Propiedades 9](#_Toc295672966)

[Métodos 9](#_Toc295672967)

[Clase TFilterIIRBS 10](#_Toc295672968)

[Diseño 10](#_Toc295672969)

[Propiedades 10](#_Toc295672970)

[Métodos 10](#_Toc295672971)

[Clase TFilterIIRLPButterworth 11](#_Toc295672972)

[Diseño 11](#_Toc295672973)

[Método 11](#_Toc295672974)

[Clase TFilterIIRLPChebyshev 12](#_Toc295672975)

[Diseño 12](#_Toc295672976)

[Propiedades 12](#_Toc295672977)

[Método 12](#_Toc295672978)

[Clase TFilterIIRLPBessel 13](#_Toc295672979)

[Diseño 13](#_Toc295672980)

[Método 13](#_Toc295672981)

[Clase TFilterIIRHPButterworth 14](#_Toc295672982)

[Diseño 14](#_Toc295672983)

[Método 14](#_Toc295672984)

[Clase TFilterIIRHPChebyshev 15](#_Toc295672985)

[Diseño 15](#_Toc295672986)

[Propiedades 15](#_Toc295672987)

[Método 15](#_Toc295672988)

[Clase TFilterIIRHPBessel 16](#_Toc295672989)

[Diseño 16](#_Toc295672990)

[Método 16](#_Toc295672991)

[Clase TFilterIIRBPButterworth 17](#_Toc295672992)

[Diseño 17](#_Toc295672993)

[Método 17](#_Toc295672994)

[Clase TFilterIIRBPChebyshev 18](#_Toc295672995)

[Diseño 18](#_Toc295672996)

[Propiedades 18](#_Toc295672997)

[Método 18](#_Toc295672998)

[Clase TFilterIIRBPBessel 19](#_Toc295672999)

[Diseño 19](#_Toc295673000)

[Método 19](#_Toc295673001)

[Clase TFilterIIRBSButterworth 20](#_Toc295673002)

[Diseño 20](#_Toc295673003)

[Método 20](#_Toc295673004)

[Clase TFilterIIRBSChebyshev 21](#_Toc295673005)

[Diseño 21](#_Toc295673006)

[Propiedades 21](#_Toc295673007)

[Método 21](#_Toc295673008)

[Clase TFilterIIRBSBessel 22](#_Toc295673009)

[Diseño 22](#_Toc295673010)

[Método 22](#_Toc295673011)

[Diseño Global de Clases 23](#_Toc295673012)

# Visión General de los Componentes

La paleta contiene 12 componentes que tienen la siguiente nomenclatura :

**Nomenclatura**

**Filter:** Indica que es un filtro.

**FilterIIR:** Indica que es un filtro de respuesta infinita al impulso.

**FilterIIRLP:** Indica que es un filtro de respuesta infinita al impulso pasa bajo.

**FilterIIRHP:** Indica que es un filtro de respuesta infinita al impulso pasa alto.

**FilterIIRBP:** Indica que es un filtro de respuesta infinita al impulso pasa banda.

**FilterIIRBS:** Indica que es un filtro de respuesta infinita al impulso supresor de banda.

**FilterIIRLPButterworth:** Indica que es un filtro de respuesta infinita al impulso pasa bajo Butterworth.

**FilterIIRLPChebyshev:** Indica que es un filtro de respuesta infinita al impulso pasa bajo Chebyshev.

**FilterIIRLPBessel:** Indica que es un filtro de respuesta infinita al impulso pasa bajo Bessel.

**FilterIIRHPButterworth:** Indica que es un filtro de respuesta infinita al impulso pasa alto Butterworth.

**FilterIIRHPChebyshev:** Indica que es un filtro de respuesta infinita al impulso pasa alto Chebyshev.

**FilterIIRHPBessel:** Indica que es un filtro de respuesta infinita al impulso pasa alto Bessel.

**FilterIIRBPButterworth:** Indica que es un filtro de respuesta infinita al impulso pasa banda Butterworth.

**FilterIIRBPChebyshev:** Indica que es un filtro de respuesta infinita al impulso pasa banda Chebyshev.

**FilterIIRBPBessel:** Indica que es un filtro de respuesta infinita al impulso pasa banda Bessel.

**FilterIIRBSButterworth:** Indica que es un filtro de respuesta infinita al impulso supresor de banda Butterworth.

**FilterIIRBSChebyshev:** Indica que es un filtro de respuesta infinita al impulso supresor de banda Chebyshev.

**FilterIIRBSBessel:** Indica que es un filtro de respuesta infinita al impulso supresor de banda Bessel.

# Units

**Filter:** Contiene propiedades y métodos que posee cualquier tipo de filtro.

**FilterIIR:** Contiene propiedades y métodos que poseen los filtros (IIR).

**FilterIIRLP:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa bajos.

**FilterIIRHP:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa altos.

**FilterIIRBP:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa bandas.

**FilterIIRBS:** Contiene propiedades y métodos que poseen los filtros (IIR) supresores de bandas.

**FilterIIRLPButterworth:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa bajo Butterworth.

**FilterIIRLPChebyshev:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa bajo Chebyshev.

**FilterIIRLPBessel:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa bajo Bessel.

**FilterIIRHPButterworth:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa alto Butterworth.

**FilterIIRHPChebyshev:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa alto Chebyshev.

**FilterIIRHPBessel:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa alto Bessel.

**FilterIIRBPButterworth:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa banda Butterworth.

**FilterIIRBPChebyshev:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa banda Chebyshev.

**FilterIIRBPBessel:** Contiene propiedades y métodos que poseen los filtros (IIR) pasa banda Bessel.

**FilterIIRBSButterworth:** Contiene propiedades y métodos que poseen los filtros (IIR) supresor de banda Butterworth.

**FilterIIRBSChebyshev:** Contiene propiedades y métodos que poseen los filtros (IIR) supresor de banda Chebyshev.

**FilterIIRBSBessel:** Contiene propiedades y métodos que poseen los filtros (IIR) supresor de banda Bessel.

**Uconst:** Contiene la declaraciones de las constates.

**UOperator:** Contieneunconjunto de funciones para trabajar con números complejos.

**UType:** Contienelas declaraciones delos tipos de datos.

# Tipos de datos

TfreqType: Tipo de dato para especificar frecuencias.

FilterOrder: Tipo de dato para el orden del filtro.

TfreqCutArray: Tipo de dato para arreglo de frecuencias.

PFreqCutArray: Tipo de dato para puntero a un arreglo de frecuencias.

TComplexArray: Tipo de dato para arreglo de números complejos.

PComplexArray: Tipo de dato para puntero a arreglo de números complejos.

TAS\_Sample: Tipo de dato para números flotantes.

TAS\_SampleArray: Tipo de dato para arreglo de números flotantes.

PTAS\_SampleArray: Tipo de dato para puntero a arreglo de números flotantes.

TAS\_Float: Tipo de dato para especificar la frecuencia de muestreo.

FilterProc: Tipo de dato para el procesamiento del valor de entrada de acuerdo a los parámetros del filtro. El resultado es  
el valor filtrado. Los datos pueden ser procesados ​​en tiempo real.

# Estructura de Clases

## Clase TFilter

La clase TFilter hereda de la clase TComponent, y constituye la clase base para creación de cualquier tipo de filtro digital, ya sea IIR ó FIR.

### Diseño

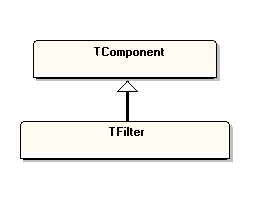


Figura 1 Clase TFilter

### Propiedades

**property** FreqCutCount : Cantidad de frecuencias de corte que tendrá el filtro.

**property** FreqCut:Frecuencia de corte.

**property** Order: Orden del filtro.

**property** SampleRate: Frecuencia de muestreo.

**property** Gain:Ganancia de la señal de salida.

**property** Info: Esta propiedad contiene información del filtro (ganancia de entrada y coeficientes de salida).

### Métodos

**procedure** Reset**; virtual**

Borra el buffer interno.

**procedure** SetupFilter(); **virtual**

Tiene la función de configurar el filtro, se declara virtual en esta clase y será redefinido en cada componente de acuerdo a sus funcionalidades.

**procedure** SetOrder(AOrder: FilterOrder)

Permite especificar el orden del filtro.

**procedure** SetSampleRate(ASampleRate: TAS\_Float)

Permite especificar la frecuencia de muestreo.

**function** FilterPassAll(InputValue: TAS\_Sample): TAS\_Sample; **virtual**

Tipo de filtro que deja pasar hacia la salida lo mismo que esta en su entrada.

**function** FilterPassNothing(InputValue: TAS\_Sample): TAS\_Sample; **virtual**

Tipo de filtro en el cual la salida siempre es 0, independientemente de la entrada.

**function** FilterFilter(InputValue: TAS\_Sample): TAS\_Sample; **virtual**

Permite filtrar la señal en función de la configuración realizada previamente.

**function** FGetInfo**:** String

Devuelve la información del filtro.

**function** GetFrequencyResponse(FreqCut1: TFreqType): TFreqType

Devuelve la respuesta del filtro en el dominio de la frecuencia.

## Clase TFilterIIR

La clase TFilterIIR hereda de la clase TFilter, y encapsula todo lo concerniente a los filtros digitales de respuesta infinita al impulso.

### Diseño

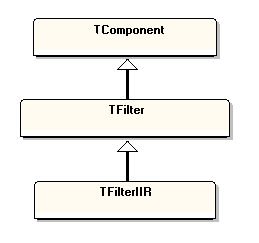


Figura 2 Clase TFilterIIR

### Propiedades

**property** ManualReset:

Esta propiedad se debe Establecer en true si desea cambiar los parámetros del filtro (frecuencias y el tipo de filtro) durante el procesamiento de datos. Por lo general, (ManualReset = false) ya que cuando los parámetros del filtro se cambian los búferes internos se borran, así se puede conseguir algunas distorsiones en el procesamiento de la señal.

**property** LastError: Descripción del error.

## Clase TFilterIIRLP

La clase TFilterIIRLP hereda de la clase TFilterIIR, y encapsula todo lo concerniente a los filtros digitales de respuesta infinita al impulso pasa bajo.

### Diseño

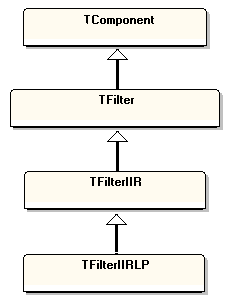


Figura 3 Clase TFilterIIRLP

### Propiedades

**property** FreqCut1: Frecuencia de corte 1.

### Métodos

**procedure** SetFreqCut1(const InputValue: TFreqType); **virtual**

Permite insertar el valor de la frecuencia de corte uno.

**procedure** ResponseLowPass

Implementa la respuesta de un filtro pasa bajo.

**function** GetFreqCut1() : TfreqType

Devuelve la frecuencia de corte uno.

## Clase TFilterIIRHP

La clase TFilterIIRHP hereda de la clase TFilterIIR, y encapsula todo lo concerniente a los filtros digitales de respuesta infinita al impulso pasa alto.

### Diseño

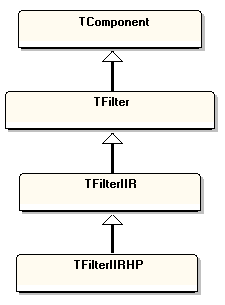


Figura 4 Clase TFilterIIRHP

### Propiedades

**property** FreqCut1: Frecuencia de corte 1.

### Métodos

**procedure** SetFreqCut1(const InputValue: TFreqType)

Permite insertar el valor de la frecuencia de corte 1.

**procedure** ResponseHighPass

Implementa la respuesta de un filtro pasa alto.

**function** GetFreqCut1() : TFreqType

Devuelve la frecuencia de corte 1.

## Clase TFilterIIRBP

La clase TFilterIIRBP hereda de la clase TFilterIIR, y encapsula todo lo concerniente a los filtros digitales de respuesta infinita al impulso pasa banda.

### Diseño

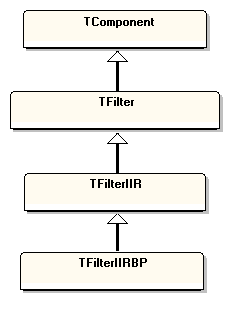


Figura 5 Clase TFilterIIRBP

### Propiedades

**property** FreqCut1 : Frecuencia de corte 1.

**property** FreqCut2 : Frecuencia de corte 2.

### Métodos

**procedure** ResponseBandPass

Implementa la respuesta de un filtro pasa banda.

**procedure** SetFreqCut1(const InputValue: TFreqType); **virtual**

Permite especificar el valor de la frecuencia de corte 1.

**procedure** SetFreqCut2(const InputValue: TFreqType); **virtual**

Permite especificar el valor de la frecuencia de corte 2.

**function** GetFreqCut1() : TFreqType

Devuelve la frecuencia de corte uno.

**function** GetFreqCut2() : TFreqType

Devuelve la frecuencia de corte 2.

## Clase TFilterIIRBS

La clase TFilterIIRBS hereda de la clase TFilterIIR, y encapsula todo lo concerniente a los filtros digitales de respuesta infinita al impulso, supresores de banda.

### Diseño

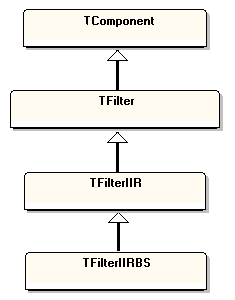


Figura 6 Clase TFilterIIRBS

### Propiedades

**property** FreqCut1 : Frecuencia de corte 1.

**property** FreqCut2 : Frecuencia de corte 2.

### Métodos

**procedure** ResponseBandStop

Implementa la respuesta de un filtro supresor de banda.

**procedure** SetFreqCut1(const InputValue: TFreqType); **virtual**

Permite especificar el valor de la frecuencia de corte 1.

**procedure** SetFreqCut2(const InputValue: TFreqType); **virtual**

Permite especificar el valor de la frecuencia de corte 2.

**function** GetFreqCut1() : TFreqType

Devuelve la frecuencia de corte 1.

**function** GetFreqCut2() : TFreqType

Devuelve la frecuencia de corte 2.

## Clase TFilterIIRLPButterworth

### Diseño

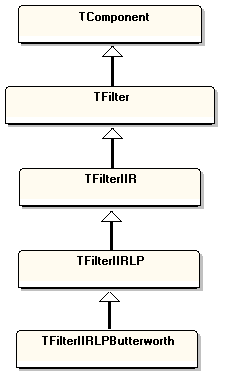


Figura 7 Clase TFilterIIRLPButterworth

### Método

Se redefine el método Setup()

## Clase TFilterIIRLPChebyshev

### Diseño

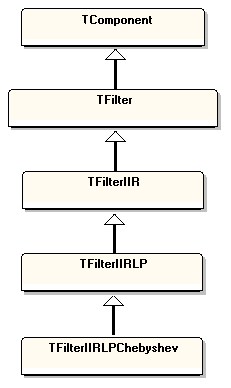


Figura 8 Clase TFilterIIRLPChebyshev

### Propiedades

**property** Ripple: La ondulación en la banda de paso en dB.

### Método

**procedure** SetRipple(ARipple: TAS\_Sample)

Permiter especificar el ripple.

**procedure** SetupForChebyshev()

Para el funcionamiento de los filtros Chebyshev.

(\*) Se redefine el método Setup()

## Clase TFilterIIRLPBessel

### Diseño

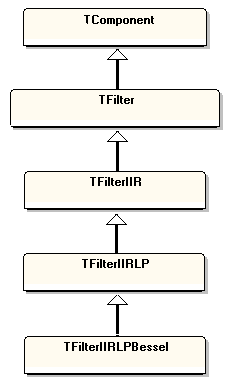


Figura 9 Clase TFilterIIRLPBessel

### Método

Se redefine el método Setup()

## Clase TFilterIIRHPButterworth

### Diseño

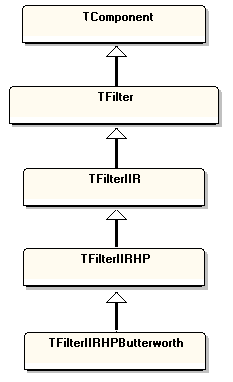


Figura 10 Clase TFilterIIRHPButterworth

### Método

Se redefine el método Setup()

## Clase TFilterIIRHPChebyshev

### Diseño

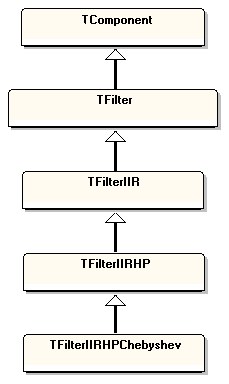


Figura 11 Clase TFilterIIRHPChebyshev

### Propiedades

**property** Ripple: La ondulación en la banda de paso en dB.

### Método

**procedure** SetRipple(ARipple: TAS\_Sample)

Permiter especificar el ripple.

**procedure** SetupForChebyshev()

Para el funcionamiento de los filtros Chebyshev.

(\*) Se redefine el método Setup()

## Clase TFilterIIRHPBessel

### Diseño

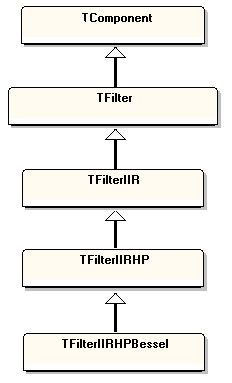


Figura 12 Clase TFilterIIRHPBessel

### Método

Se redefine el método Setup()

## Clase TFilterIIRBPButterworth

### Diseño

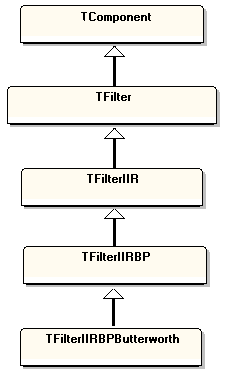


Figura 13 Clase TFilterIIRBPButterworth

### Método

Se redefine el método Setup()

## Clase TFilterIIRBPChebyshev

### Diseño

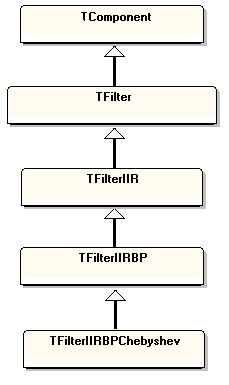


Figura 14 Clase TFilterIIRBPChebyshev

### Propiedades

**property** Ripple: La ondulación en la banda de paso en dB.

### Método

**procedure** SetRipple(ARipple: TAS\_Sample)

Permiter especificar el ripple.

**procedure** SetupForChebyshev()

Para el funcionamiento de los filtros Chebyshev.

(\*) Se redefine el método Setup()

## Clase TFilterIIRBPBessel

### Diseño

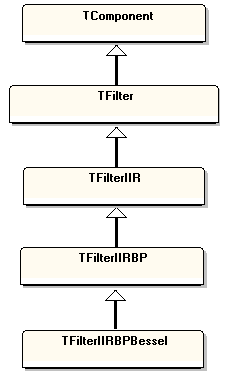


Figura 15 Clase TFilterIIRBPBessel

### Método

Se redefine el método Setup()

## Clase TFilterIIRBSButterworth

### Diseño

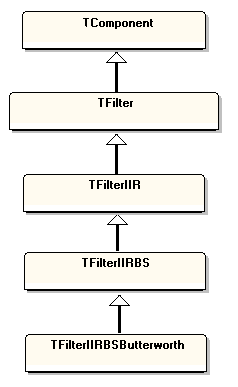


Figura 16 Clase TFilterIIRBSButterworth

### Método

Se redefine el método Setup()

## Clase TFilterIIRBSChebyshev

### Diseño

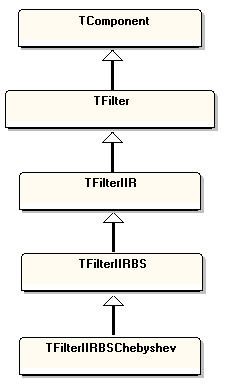


Figura 17 Clase TFilterIIRBSChebyshev

### Propiedades

**property** Ripple: La ondulación en la banda de paso en dB.

### Método

**procedure** SetRipple(ARipple: TAS\_Sample)

Permiter especificar el ripple.

**procedure** SetupForChebyshev()

Para el funcionamiento de los filtros Chebyshev.

(\*) Se redefine el método Setup()

## Clase TFilterIIRBSBessel

### Diseño

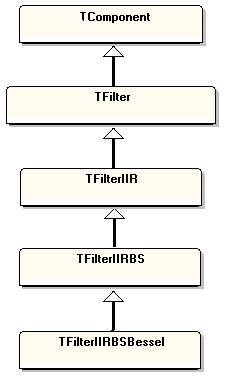


Figura 18 Clase TFilterIIRBSBessel

### Método

Se redefine el método Setup()

# Diseño Global de Clases

