# Descripción General

Este módulo contiene toda la interfaz visual que se utiliza en el sistema duetto Sound Lab. Su programación se basa principalmente en la utilización de las clases del framework PyQt para la manipulación de los eventos, ventanas, diálogos y funcionamiento general. La librería pyqtgraph es utilizada para la implementación de varios de los controles encargados de graficar las señales que integran el sistema así como para redefinir algunas clases de PyQt.

Módulo Graphic\_Interface

Documentación para el programador

# Sub-módulos

El módulo Graphic Interface publica los siguientes módulos:

* Dialogs 🡪 Las ventanas de diálogo que se muestran en las distintas funcionalidades del sistema.
* UI Files 🡪Los ficheros .ui que se generan por el framework PyQt para la generación de ventanas.
* Widgets 🡪 Los distintos controles visuales que se implementan o redefinen en la aplicación.
* Windows 🡪 Las distintas ventanas que componen el sistema.

A continuación se profundizará en cada módulo.

# Módulo Dialogs

## **Descripción General**

Este módulo define todas las ventanas de diálogos que se utilizan en el sistema para el ingreso de parámetros a las funcionalidades.

## **Clases**

1. Change Volume 🡪 parámetros para la modificación de volumen o amplitud.
2. Filter 🡪 parámetros para el filtrado de la señal en el dominio de la frecuencia.
3. Insert Silence 🡪 parámetros para insertar silencio.
4. New File 🡪 parámetros para la creación de una nueva señal.
5. ElemDetectSettingsDialog 🡪 parámetros para la detección de segmentos.

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

A excepción de ElemDetectSettingsDialog el resto de los diálogos solo son utilizados como simple vía de ingreso de parámetros y no poseen ninguna dificultad en su implementación.

## **Clase ElemDetectSettingsDialog**

Descripción

Este diálogo permite la definición de los parámetros que se utilizarán para la segmentación de la señal. Permite ….

***Variables***

***Métodos***

***Invariantes, Reglas y Convenciones***

# Módulo UI Files

## **Descripción General**

En este módulo se almacenan los recursos (íconos) de la aplicación y los ficheros .ui de las ventanas del sistema. No se relaciona directamente con el resto de la aplicación y puede ser excluido del producto final. Solo se incluye como parte de este módulo de interfaz visual con fines organizativos.

# Módulo Widgets

## **Descripción General**

En este módulo se ubican todos los controles visuales desarrollados como parte de la implementación del sistema. Se asume la programación de los controles como modular e independiente aunque existen dependencias entre ellos para su uso en el software. En principio cada control es posible reutilizarlo y extenderlo para dotarlo de nuevas funcionalidades.

## **Clases**

1. HorizontalHistogramWidget 🡪 extiende el control histogram de pyqtgraph para su visualización de forma horizontal.
2. HorizontalHistogramItem 🡪item relacionado con el control anterior.
3. SpectrogramPlotWidget 🡪 control que grafica el espectrograma. Se basa en reutilización de clases de pyqtgraph
4. SpecYAxis 🡪 clase que se integra con el control anterior.
5. OscilogramPlotWidget 🡪 control que grafica el oscilograma. Extiende la funcionalidad del control PlotWidget de pyqtgraph
6. QSignalVisualizerWidget 🡪 es el control más importante del sistema ya que integra los demás controles y brinda la mayoría de las funcionalidades. Gran parte del procesamiento realizado por la aplicación resulta en wrappers a métodos en este control.
7. OscXAxis 🡪clase que se integra con el control anterior.
8. OscYAxis 🡪 clase que se integra con el control anterior
9. Tools 🡪 enum que define las herramientas que los usuarios podrán utilizar en el sistema. Se utiliza en el control anterior y en los controles OscilogramPlotWidget y SpectrogramPlotWidget.
10. RectROI 🡪 ?
11. UndoRedoManager,UndoRedoActions 🡪 clases para proveer al control de la funcionalidad de undo y redo para las acciones de edición sobre la señal.

## **Clase HorizontalHistogramWidget**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase HorizontalHistogramItem**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

## **Clase SpectrogramPlotWidget**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase SpecYAxis**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase OscilogramPlotWidget**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase QSignalVisualizerWidget**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase OscXAxis**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase OscYAxis**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase RectROI**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase UndoRedoManager**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase UndoRedoAction**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

# Módulo Windows

## **Descripción General**

En este módulo se incluyen todas las ventanas que se utilizan en el sistema. Las ventanas de procesamiento están directamente vinculadas con los controles del módulo anterior y se relacionan con ellos para brindar las funcionalidades.

## **Clases**

1. DuettoSoundLabWindow 🡪 ventana principal de la aplicación. Procesamiento y edición de señales.
2. TwoDimensionalWindow 🡪 ventana de visualización de procesamientos bidimensionales.
3. SegmentationAndClasificationWindow 🡪 Ventana que permite segmentar y clasificar la señal.
4. SerializedData 🡪 estructura para hacer persistente a disco las opciones visuales del sistema para una futura utilización.
5. DuettoListParameterItem 🡪 clase para la ordenacion de los parámetros de algunas opciones visuales en la ventana DuettoSoundLabWindow

## **Clase DuettoSoundLabWindow**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase TwoDimensionalWindow**

Descripción

Esta ventana es utilizada en la ventana de segmentación y clasificación. Luego de segmentar y extraer parámetros. La ventana permite representar y graficar bidimensionalmente dos de las magnitudes medidas. Se utiliza un scatter plot para el gráfico de los elementos. La ventana permite analizar los segmentos que se detectaron en la ventana **SegmentationAndClasificationWindow.** Mediante esta ventana se puede visualizar los parámetros medidos en cada segmento. Contiene un Parameter Tree en el que se insertan las opciones que brinda la ventana. Allí se puede encontrar las dos listas de parámetros (cada una con las mediciones realizadas) en las cuales se pueden seleccionar cual parámetro se graficará en cada eje (X e Y). Se pueden modificar de la misma manera algunas opciones visuales como la figura de los elementos bidimensionales (cuadrados, rombos etc), el color, la fuente para el texto de los ejes etc.

***Variables***

1. elementSelected (signal) 🡪 Esta señal es lanzada cuando se selecciona un elemento bidimensional del gráfico. Es la primera parte de la comunicación entre la ventana de segmentación y esta ventana. Permite avisar cuando se selecciona un elemento desde esta ventana.
2. elementsClasification (signal) 🡪 Esta señal es lanzada cuando se han seleccionado múltiples objetos del gráfico y se les han asignado categorías dentro de la clasificación. Es la implementación de la clasificación manual.
3. scatter\_plot (ScatterPLot) 🡪 Es el grafico que se muestra. Está formado por tantos elementos como hayan sido detectados en la ventana de segmentación.
4. classificationData (ClasificationData) 🡪 Es la instancia de clasificación. En ella se almacena las categorías, los valores de cada categoría, así como los vectores de entrenamiento que son comunes a los métodos de clasificación. Es utilizada para la segmentación manual en esta ventana.
5. Columns (string[]) 🡪 Los nombres de los parámetros medidos.
6. previousSelectedElement (int) 🡪 El índice del último elemento seleccionado. Es utilizado para resaltar con otro color el elemento seleccionado así como para reorganizar visualmente la ventana de segmentación con el objetivo de hacerlo visible en la tabla de parámetros y en el control visual de oscilograma y espectrograma. Cuando se selecciona mediante click un elemento este valor es actualizado y es lanzada la señal elementSelected
7. data (array[,]) 🡪 Array de (cantidad de segmentos)x(cantidad de parámetros medidos).

***Métodos***

1. plot 🡪 refresca el gráfico bidimensional con los valores contenidos en data. Genera un elemento por cada fila en data y lo posiciona en el grafico según las coordenadas seleccionadas por el usuario. Las coordenadas dependen del parámetro seleccionado para cada eje.
2. on\_actionSaveGraphImage\_triggered 🡪 Permite exportar la imagen del grafico a disco para su futuro uso.
3. on\_actionMark\_Selected\_Elements\_As\_triggered 🡪 Luego de seleccionar varios elementos visuales en la ventana (mediante dibujo de un rectángulo. Haciendo click para fijar la posición inicial y haciendo drag para fijar las dimensiones alto y ancho) este método procede a abrir el dialog de clasificación manual para que el usuario fije los valores de cada categoría para todos los elementos que se hayan seleccionado.
4. on\_actionHide\_Show\_Settings\_triggered 🡪 Permite seconder y volver a mostrar el Parameter tree con las opciones. Es conveniente si el usuario quiere observer con mayor comodidad el grafico.
5. createParameterTreeOptions Método privado que crea el parameter tree con los datos de las mediciones (posible valores para cada eje).
6. loadData (columns(string[]), data (array[,])) 🡪Método que actualiza los valores de la ventana porque hubo algún cambio en las mediciones o en los segmentos detectados.
7. load\_Theme 🡪 Carga con las opciones visuales de la aplicación en esta nueva ventana
8. selectElement (index(int)) 🡪 Permite seleccionar el elemento que se encuentra en la posición index. Es la segunda parte de la comunicación entre la ventana de segmentación y esta.
9. deselectElement 🡪 El complemento del método anterior para revertir la selección

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase SegmentationAndClasificationWindow**

Descripción

En esta ventana se realiza la segmentación, clasificación y extracción de parámetros sobre la señal. Brinda las funcionalidades de la ventana principal que no modifican la señal. Contiene un control SignalDetectorWidget para la visualización de la señal. Este control extiende las funcionalidades del SignalVisualizer para la segmentación y visualización de segmentos.

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase SerializedData**

Descripción

Esta clase es la utilizada para serializar el tema visual utilizado en la aplicación.

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

## **Clase DuettoListParameterItem**

Descripción

***Variables***

1. …

***Métodos***

1. …

***Invariantes, Reglas y Convenciones***

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****