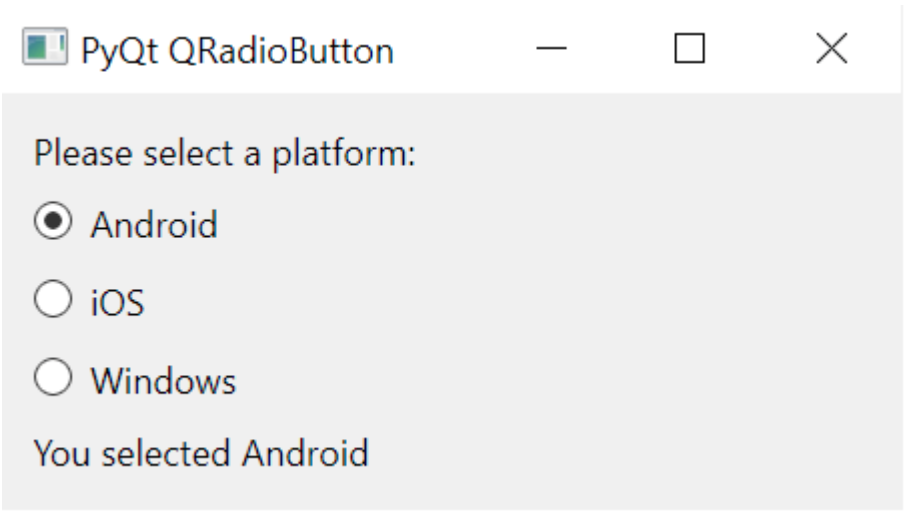
•QRadioButton

Uso principal→ Es un botón que se puede activar y desactivar. El usuario tiene que elegir entre varias opciones y solo puede marcar una.

Este botón suele mostrar texto y un pequeño icono (la zona donde marcar o desmarcar).

Imagen de ejemplo



Algunos métodos

- ✓ .setChecked(bool) → método para establecer el estado de una casilla de verificación, marcado (true) y desmarcado(false).
- ✓ .isChecked() →es un booleano y sirve para comprobar si un botón está seleccionado o no, es decir, sirve para comprobar su estado. Por defecto, el botón esta desmarcado.
- ✓ .text() → contiene el texto que se muestra en el botón, si el botón no tiene texto esta función devolverá una cadena vacia.

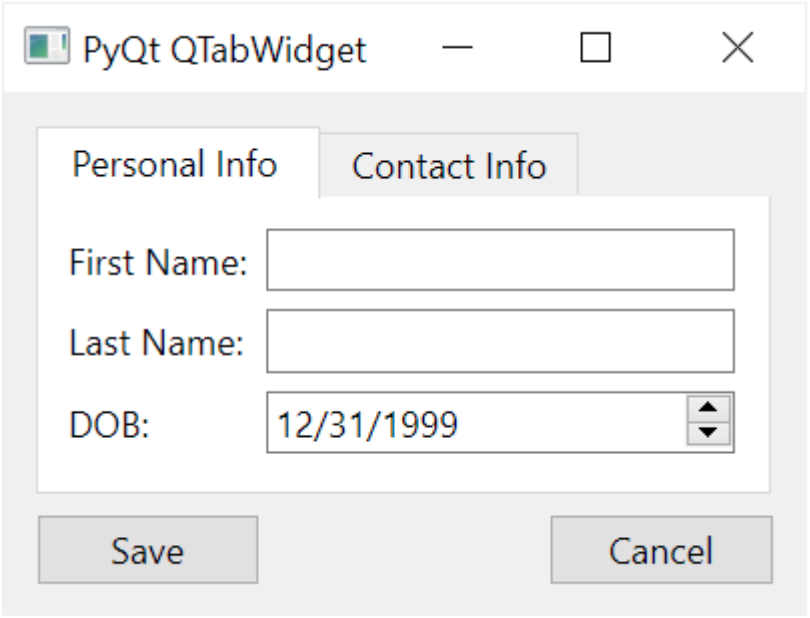
Señales más frecuentes

toggled(bool) → señal que se emite cada vez que el estado del botón cambia, de marcado a desmarcado o de desmarcado a marcado Esta señal se utiliza para conectar el cambio de estado con otras partes de la aplicación, para realizar un acción concreta en respuesta a la selección del usuario.

•QTabWidget

Uso principal→ widget de pestaña que proporciona una barra de pestañas y un área que se utiliza para mostrar las páginas relacionadas con cada pestaña. Cada pestaña está asociado a un widget diferente. Es decir, permite gestionar una colección de widgets con pestañas en una interfaz gráfica.

Imagen de ejemplo:



Algunos métodos

- ✓ .addTab(widget, "Título") → agrega una nueva pestaña al widget. La versión básica es add(página,texto), donde página es un QWidget que contendrá el contenido de la pestaña y texto (en este caso titulo) es un QString con el titulo de la pestaña.
- ✓ .setCurrentIndex(int) → esta propiedad contiene la posición del índice de la página de la pestaña actual y permite seleccionarla, si no hay widget, su índice será -1.
- ✓ .count() → esta propiedad almacena el número de pestañas que hay en la barra de pestañas. Por defecto, su valor es 0.

Señales más frecuentes

currentChanged(int) → señal que se emite cada vez que cambia el índice de la página actual.

tabBarClicked(int) → señal que se emite cuando el usuario hace clic en una pestaña

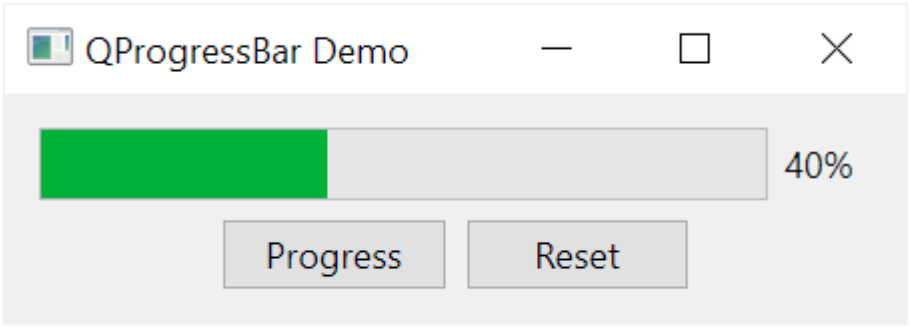
tabBarBodubleClicked(int) → señal que se emite cuando el usuario hace doble click en una pestaña

•QProgressBar

Uso principal→ sirve para comunicar el avance de un proceso o tarea al usuario.

Su objetivo es dar información visual sobre el estado actual y el tiempo restante para completar una acción, es decir, ofrecer información visual clara y concisa sobre el estado de una operación que está realizando al usuario.

Imagen de ejemplo



Algunos métodos

- ✓ .setValue(int) → almacena el valor actual de la barra de progreso.
- ✓ .setRange(min, max) → establece los valores mínimos y máximos de la barra de progreso.
- ✓ .reset() → reinicia la barra de progreso.
- ✓ setOrientation → esta propiedad controla la orientación de la barra de progreso. La orientación debe ser horizontal o vertical.

Señales más frecuentes

•QDateTimeEdit

Uso principal→ es una combinación entre QDate y Qtime, es decir, contiene tanto la fecha como la hora por lo que permite editar fechas y horas mediante el teclado o las teclas de flecha para aumentar o disminuir estos valores.

•QSlider

•QDial

https://doc.qt.io/qtforpython-6/

https://www.pythontutorial.net/pyqt/

https://ftp.nmr.mgh.harvard.edu/pub/dist/freesurfer/tutorial_packages_centos6/centos6/freesurfer-fsl-matlab-Linux-centos6_x86_64-dev/freesurfer/lib/qt/qt_doc/html/qabstractbutton.html#checked-prop

https://runebook.dev/es/docs/qt/qtabwidget#details

https://web.mit.edu/~firebird/arch/sun4x_58/doc/html/qtabwidget.html#count

https://www.uxables.com/diseno-ux-ui/disenando-barras-de-progreso-progress-bar/