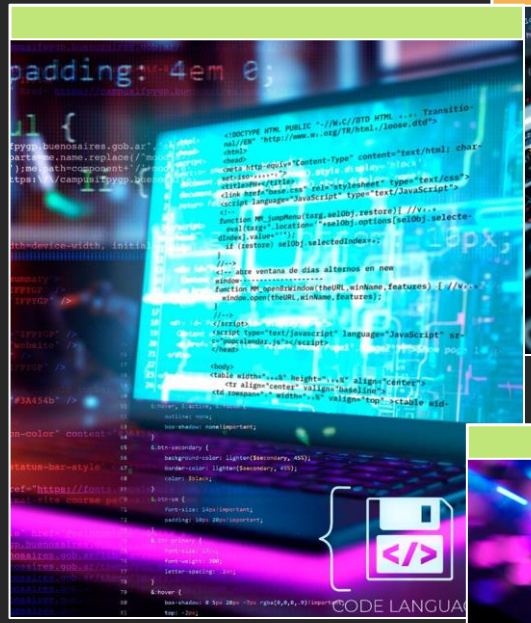
Programacion II [PRACTICA]

1 0 0 1 0 1 0 1 0 1 0 1 0 0

1 0 1 0 0 1 0 1 0 1 0 0 1 0 1 0

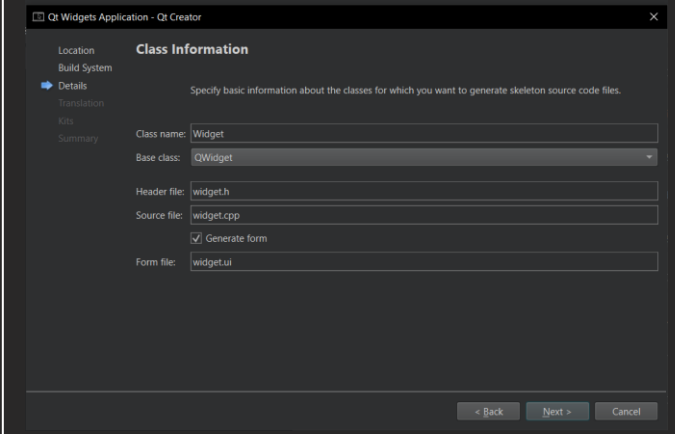
INTERFAZ GRAFICA DE USUARIOS (G.U.I)



1 0 1 0 1 0 1 1 1 0 0 0 1 1 1 0

1 1 0 0 1 0 1 0 1 0 1 0 1 1 1 0

Tipos de Ventanas - QWidget



QWidget es la clase base de todos los objetos de interfaz de usuario en Qt. Representa un contenedor o ventana vacía que puede contener otros widgets, pero no tiene ninguna estructura o funcionalidad específica predefinida, como barras de menú o barras de herramientas.

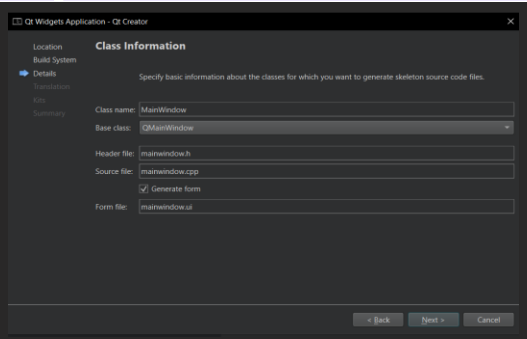
Características:

- **Minimalista:** No incluye funcionalidades adicionales como barras de menú, barras de estado, barras de herramientas, etc.
- **Flexible:** Se puede utilizar como una ventana o como un widget dentro de otros contenedores. Ideal para crear widgets personalizados o ventanas sencillas.
- **Uso como contenedor:** Es muy común utilizar QWidget como contenedor básico para otros widgets en layouts complejos.

Métodos Comunes:

- **resize(int width, int height):**
Cambia el tamaño del
- **widget.move(int x, int y):**
Mueve el widget a una nueva
- **posición.show():**
Muestra el widget en pantalla.
- **setWindowTitle(const QString &title):**
Establece el título de la ventana.

Tipos de Ventanas - QMainWindow

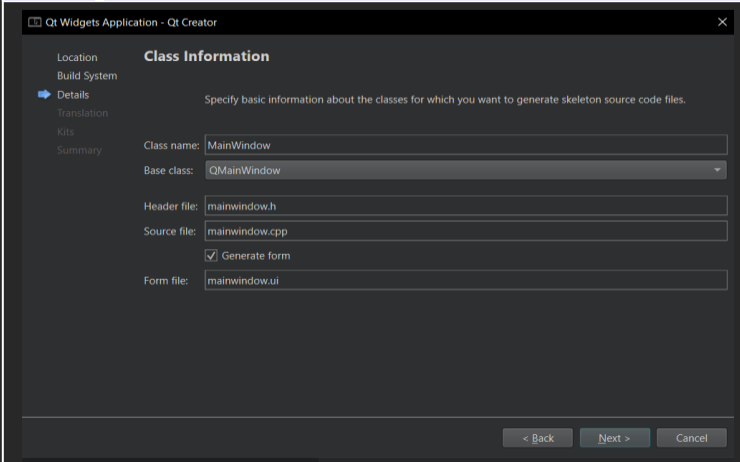


QMainWindow es una subclase de QWidget diseñada específicamente para crear aplicaciones de ventana principal. Ofrece una estructura de ventana estándar con funcionalidades comunes como una barra de menú, una barra de herramientas, una barra de estado y una área central para el contenido principal.

Características:

- **Estructura estándar**: Proporciona una estructura típica de aplicación con componentes predefinidos:
  - **Menu Bar (menuBar()):** Una barra de menú para agregar elementos como "Archivo", "Editar", etc.
  - **Tool Bar (addToolBar()):** Barras de herramientas para agregar botones y acciones.
  - **Status Bar (statusBar()):** Una barra de estado para mostrar información contextual.
  - **Central Widget (setCentralWidget()):** Un widget principal que ocupa el espacio central de la ventana.
- **Ventana completa**: Ideal para aplicaciones que requieren una interfaz compleja con varios componentes.
- **Acciones y Menús**: Permite agregar acciones y menús de forma sencilla.

Tipos de Ventanas - QMainWindow

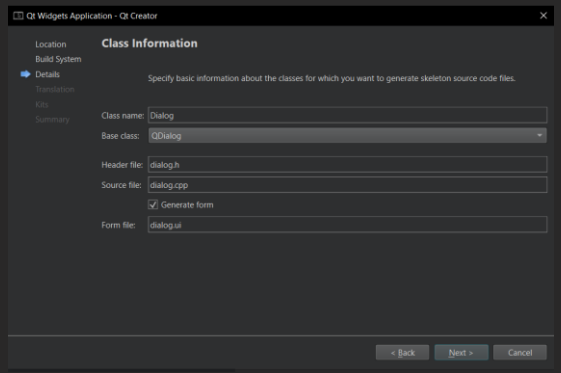


QMainWindow es una subclase de QWidget diseñada específicamente para crear aplicaciones de ventana principal. Ofrece una estructura de ventana estándar con funcionalidades comunes como una barra de menú, una barra de herramientas, una barra de estado y una área central para el contenido principal.

Métodos Comunes:

- **menuBar():** Devuelve la barra de menú de la ventana principal.
- **addToolBar(const QString &title):** Añade una barra de herramientas con un título específico.
- **setCentralWidget(QWidget *widget):** Establece el widget central de la ventana.
- **statusBar():** Devuelve la barra de estado para mostrar mensajes.

Tipos de Ventanas - QDialog

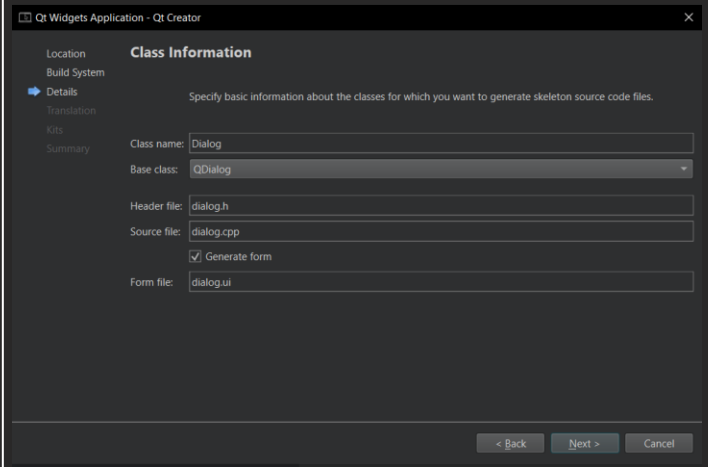


QDialog es una subclase de QWidget diseñada específicamente para representar ventanas de diálogo modales o no modales. Su propósito principal es interactuar con el usuario para solicitar información o proporcionar mensajes.

Características:

- **Modales:** Un QDialog puede ser modal o no modal. Un diálogo modal bloquea la interacción con otras ventanas de la aplicación hasta que el diálogo se cierra. Esto es útil para diálogos que requieren la atención del usuario antes de continuar.
- **Ejecución Síncrona:** Tiene métodos como exec() que bloquean el flujo del programa hasta que el diálogo se cierra, devolviendo un valor que indica cómo se cerró el diálogo (Accepted o Rejected).
- **Botones estándar:** Los diálogos suelen tener botones predefinidos como "OK", "Cancel", "Yes", "No", etc. Esto facilita la creación de diálogos con respuestas típicas.
- **Personalizable:** Aunque su estructura está orientada a interacciones específicas, se puede personalizar añadiendo widgets adicionales para recoger datos del usuario.

Tipos de Ventanas - QDialog



QDialog es una subclase de QWidget diseñada específicamente para representar ventanas de diálogo modales o no modales. Su propósito principal es interactuar con el usuario para solicitar información o proporcionar mensajes.

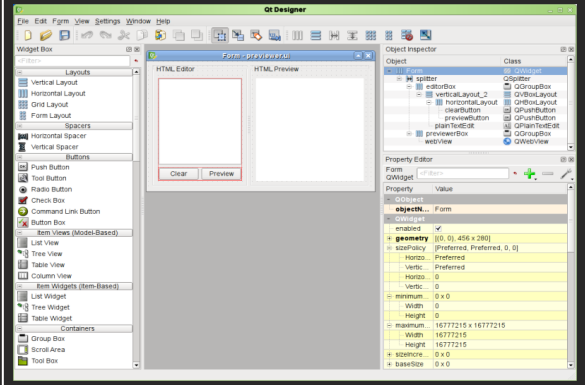
Métodos Comunes:

- **exec()**: Muestra el diálogo de manera modal y bloquea el resto de la aplicación hasta que se cierra. Devuelve QDialog::Accepted o QDialog::Rejected.
- **show()**: Muestra el diálogo de manera no modal.
- **accept()**: Cierra el diálogo y devuelve QDialog::Accepted.
- **reject()**: Cierra el diálogo y devuelve QDialog::Rejected.
- **setModal(bool)**: Establece si el diálogo debe ser modal o no.

Tabla Comparativa – Tipos de Ventana

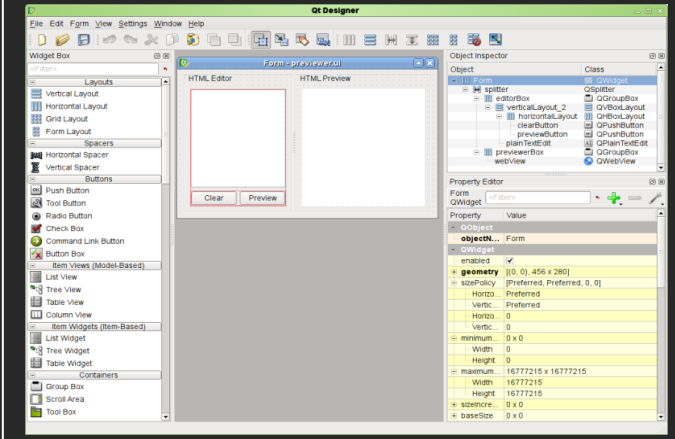
Características	QWidget	QMainWindow	QDialog
Propósito	Contenedor básico para otros widgets.	Ventana principal de la aplicación con estructura estándar.	Ventana de diálogo para interacciones específicas.
Estructura	Simple, sin componentes adicionales.	Incluye barra de menú, barra de herramientas, barra de estado, y widget central.	Orientada a interacción, con botones para confirmar o cancelar.
Uso Principal	Crear widgets personalizados o ventanas sencillas.	Crear la ventana principal de aplicaciones complejas.	Crear ventanas modales o no modales para mensajes, confirmaciones o recolección de datos.
Modalidad	No tiene modalidad intrínseca.	No tiene modalidad intrínseca.	Puede ser modal (bloquea la interacción con otras ventanas).
Métodos de Visualización	show(), hide()	show(), hide()	exec() (modal), show() (no modal), accept(), reject().
Ejecución Síncrona	No.	No.	Sí, con exec() bloquea el resto de la aplicación hasta cerrarse.
Barra de Menú	No integrada.	Sí, con menuBar().	No integrada, pero se puede agregar manualmente.
Barra de Herramientas	No integrada.	Sí, con addToolBar().	No integrada, pero se puede agregar manualmente.
Barra de Estado	No integrada.	Sí, con statusBar().	No integrada.
Widget Central	No tiene un widget central predefinido.	Sí, con setCentralWidget().	No tiene un widget central predefinido.
Diseño Flexible	Sí, se puede usar para cualquier tipo de interfaz.	Estructura rígida pero con componentes útiles para aplicaciones complejas.	Sí, se puede personalizar para diferentes tipos de diálogos.
Tipo de Aplicación	Widgets individuales, ventanas simples.	Aplicaciones de escritorio con interfaz completa.	Ventanas secundarias para interacciones con el usuario.
Ejemplos de Uso	Widgets personalizados, contenedores.	Editores de texto, navegadores, aplicaciones con múltiples componentes.	Confirmaciones, mensajes, formularios breves.

QT Design - QObjets



- **Layouts:** Utilizados para organizar widgets dentro de un contenedor. Los más comunes son los layouts verticales (QVBoxLayout), horizontales (QHBoxLayout), en cuadrícula (QGridLayout), y de formulario (QFormLayout).
- **Spacers:** Elementos que se utilizan para agregar espacio vacío en los layouts, permitiendo organizar los widgets con mayor flexibilidad.
- **Buttons:** Botones de diferentes tipos (QPushButton, QRadioButton, QCheckBox) que permiten al usuario realizar acciones o seleccionar opciones.
- **Item Views:** Vistas para mostrar datos en diferentes formatos, como listas (QListView), tablas (QTableView), árboles (QTreeView), y columnas (QColumnView).
- **Item Widgets:** Widgets que integran funcionalidad de elementos con soporte para elementos personalizados, como listas (QListWidget), tablas (QTableWidget), y árboles (QTreeWidget).

QT Design - QObjets



- **Containers:** Contenedores que organizan otros widgets, como QGroupBox para agrupar widgets relacionados, QTabWidget para organizar en pestañas, y QStackedWidget para mostrar un widget a la vez.

- **Input Widgets:** Widgets que permiten la entrada de datos del usuario, como QLineEdit para texto de una sola línea, QComboBox para listas desplegables, y QSlider para valores deslizables.

- **Display Widgets:** Widgets para mostrar información, como QLabel para texto o imágenes, QLCDNumber para números en estilo de pantalla LCD, y QProgressBar para barras de progreso.

Tabla Comparativa - QObjets

Categoría	Componente	Descripción	Métodos Esenciales
Layouts	QVBoxLayout	Layout vertical que organiza widgets de arriba hacia abajo.	addWidget(), addLayout(), setSpacing(), setMargin()
	QHBoxLayout	Layout horizontal que organiza widgets de izquierda a derecha.	addWidget(), addLayout(), setSpacing(), setMargin()
	QGridLayout	Layout en forma de tabla que organiza widgets en filas y columnas.	addWidget(), addLayout(), setRowStretch(), setColumnStretch()
	QFormLayout	Layout para formularios con etiquetas y campos alineados.	addRow(), setLabelAlignment(), setFieldGrowthPolicy()
Spacers	QSpacerItem	Espaciador que se usa para agregar espacio vacío en un layout.	sizeHint(), setSizePolicy()
Buttons	QPushButton	Botón estándar que puede ejecutar una acción cuando se presiona.	setText(), setIcon(), clicked()
	QRadioButton	Botón de opción que permite seleccionar una opción de un grupo.	setChecked(), toggled()
	QCheckBox	Caja de verificación que permite seleccionar o deseleccionar una opción.	setChecked(), toggled()
	QToolButton	Botón de herramientas con icono, utilizado en barras de herramientas.	setIcon(), setPopupMode(), setMenu()
Item Views	QListView	Vista de lista que muestra elementos en una lista vertical.	setModel(), setSelectionMode(), setSpacing()
	QTableView	Vista de tabla para mostrar datos en formato de filas y columnas.	setModel(), setSelectionMode(), setColumnWidth()
	QTreeView	Vista de árbol para mostrar datos jerárquicos con elementos expandibles.	setModel(), setSelectionMode(), setColumnWidth()
	QColumnView	Vista de columnas para mostrar datos en una jerarquía.	setModel(), setSelectionMode(), setColumnWidth()

Retos de Programacion

01 Reloj Digital con QLCDNumber

Crea una aplicación usando QWidget que muestre un reloj digital en tiempo real.

- Utiliza un QLCDNumber para mostrar la hora actual en formato "hh:mm"
- Un QTimer para actualizar el reloj cada segundo.
- Incluye un botón "Detener" para pausar la actualización del reloj.

Objetivos: Practicar con QWidget, QLCDNumber, QTimer, y QPushButton.

1 0 0 1 0 1 0 1 0 0 1 0 1 0 1 0

1 0 1 0 0 1 0 0 1 0 1 1 0 1 1 1

02 Configurador de Preferencias con QDialog no Modal

Implementa un configurador de preferencias que aparezca en un QDialog no modal. El diálogo debe permitir seleccionar:

- El color del fondo de la aplicación (usando QComboBox).
- El tamaño de la fuente (usando QSpinBox).
- Un botón "Aplicar" que cambie el fondo y el tamaño de la fuente en la ventana principal.

La ventana principal debe ser un QWidget con un texto de ejemplo que refleje las preferencias seleccionadas.

Objetivos: Usar un QDialog no modal, QComboBox, QSpinBox, y conectar los cambios con un QWidget principal.

03 Aplicación de Conversión de Unidades

Implementa una aplicación para convertir temperaturas entre Celsius y Fahrenheit. La ventana principal debe ser un QWidget que contenga:

- Un QLineEdit para ingresar una temperatura.
- Dos QRadioButton: uno para seleccionar la conversión de Celsius a Fahrenheit y otro para Fahrenheit a Celsius.
- Un botón "Convertir" que realice la conversión seleccionada.
- Un QLabel que muestre el resultado de la conversión.

Objetivos: Practicar con QWidget, QLineEdit, QRadioButton, QPushButton, y QLabel para construir una interfaz interactiva de conversión.

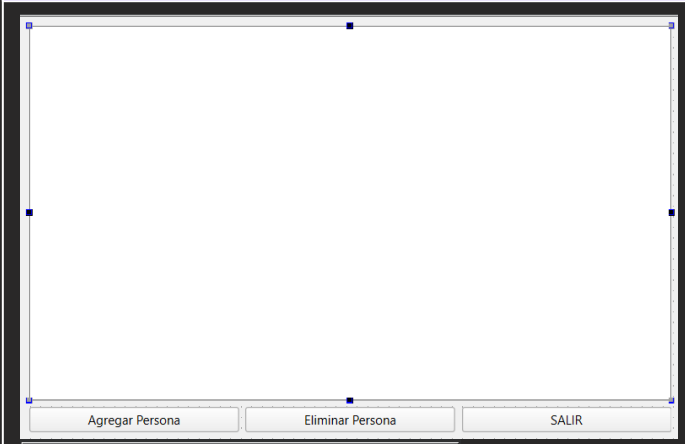
04 Simulador de Temporizador con QProgressBar

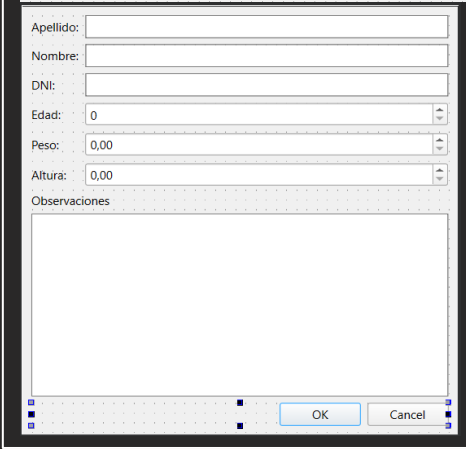
Implementa una aplicación de temporizador que muestre el progreso en una QProgressBar.
La ventana principal debe ser un QWidget con:

- Un QSpinBox para seleccionar el tiempo en segundos.
- Un botón "Iniciar Temporizador" que inicie el temporizador.
- Una QProgressBar que muestre el progreso del temporizador.
- Un QLabel que muestre el tiempo restante en segundos.
- Usa QTimer para actualizar el temporizador cada segundo.

Objetivos: Practicar con QWidget, QProgressBar, QSpinBox, QTimer, y QLabel.

05 Formularios





Crear el siguiente formulario en una ventana del tipo QDialog [Solo Front]
El programa debera tener:

- QMainWindow
  - 01 Qlabel
  - 01 QLCDDisplay
- Qwiget:
  - QTable
  - Botón Agregar
  - Botón Eliminar
  - Botón SALIR
- Qdialog:
  - 07 QLabel
  - 03 QLineEdit
  - 01 QSpinBox
  - 02 QDouble SpinBox
  - 01 QPlainTextEdit
  - 02 botones (OK/CANCEL)

1 0 1 0 0 1 0 0 1 0 1 1 0 1 1 1