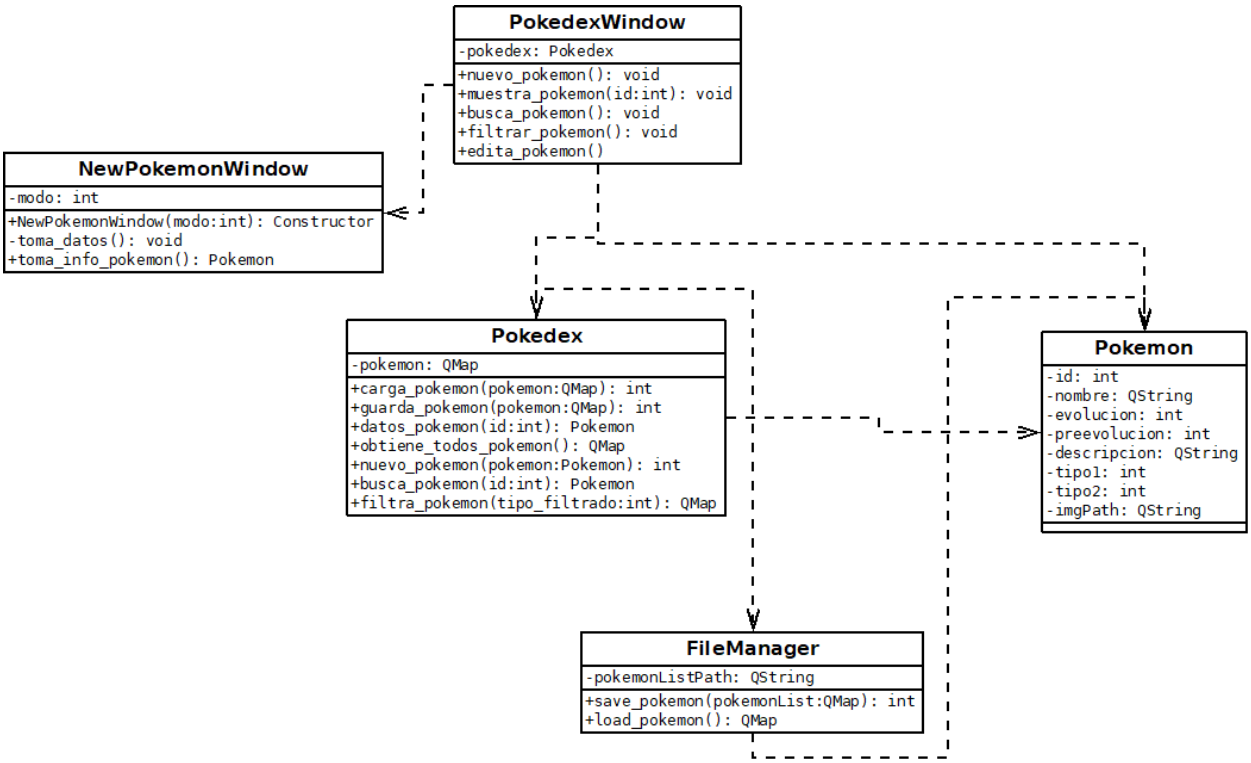
# ¡Implementando una mini pokedex!

Se deberá implementar un programa el cual sirva para registro de datos basados en Pokemon (en los juegos conocido como pokedex, en las cuales vas registrando los datos de cada uno de estos).

Se necesitará implementar todo esto en el lenguaje de C++ utilizando el framework de QT para implementar la interfaz gráfica, a su vez, está prohibido utilizar el QT Designer, todo deberá de ser implementado utilizando código.

Deberá tener las siguientes clases, atributos y funciones base de acuerdo con el diagrama UML mostrado, más aparte lo que necesiten ustedes para llevar a cabo el programa.




Estará compuesto de la siguiente manera:

Clase PokedexWindow: Esta clase va a ser su ventana principal, por lo que esta deberá heredar de QMainWindow y deberá de ser la ventana que se muestre al arrancar el programa.

Variables:

- pokedex: de tipo Pokedex, este objeto es el que se va a encargar de manejar realmente todos los movimientos solicitados hacia la pokedex, siendo PokedexWindow la clase que muestre los datos, que cada vez que el usuario solicite algo por medio de PokedexWindow, esta mande a

llamar a su objeto pokedex para que haga el trabajo, regresarnos un resultado y poder mostrarlo por medio del mismo PokedexWindow.

Funciones:

- nuevo_pokemon(): Esta función debe de abrir una ventana de tipo NewPokemonWindow, en la cual, antes de, se deberá tener que ingresar por medio de un widget en la interfaz si se necesita ingresar en modo nuevo en modo editar, en seguida lanzar un exec para esperar a que cierren la ventana y al final, en caso de que hayan colocado OK, tomar los datos obtenidos y guardarlos en la pokedex por medio del objeto del mismo nombre.
- muestra_pokemon(): Esta función debe ser un slot, el cual si se selecciona un pokemon en la lista de pokemon registrados, se deben de mostrar en un apartado, todos los datos registrados del pokemon seleccionado, claro, pidiendo los datos al objeto pokedex.
- busca_pokemon(): Por medio de un widget lineedit, se podrá escribir una cadena en la cual, se buscará coincidencias con los nombres de los pokemon que tengamos registrados en la pokedex, actualizando la lista de pokemon visible con todas las coincidencias encontradas. Si se borra, también deberá de seguir actualizando la lista, todo esto pidiendo al objeto pokedex que te haga ese trabajo. Esta función puede ser un slot.
- filtrar_pokemon(): Se deberá tener un widget en el cual se pueda hacer un filtrado, este puede ser un combox(widget utilizado para utilizar múltiples opciones como el select/option de HTML), en el cual se van a filtrar por tipo la lista de pokemon de la pokedex mostrada, si no se pide tipo alguno, se deberá mostrar la lista completa de pokemon registrados, todo esto pidiéndole los datos al objeto pokedex. Serán 2 filtros, ambos por tipo de pokemon, pero uno para tipo 1 y otro para el tipo 2 que se carga.
- edita_pokemon(): Esta opción será casi igual que la de nuevo_pokemon(), solo que en esta deberás de entrar en modo editar, pasandole el objeto de Pokemon a editar hacia la nueva ventana y tener esta referencia para guardar los datos en donde se deben.

Clase Pokedex: Esta va a ser la clase que se va a encargar realmente de administrar todos los datos que se tengan de la pokedex, pasando por ella cualquier tipo de movimientos posibles para el funcionamiento correcto del programa, guardar, cargar, almacenar, eliminar, filtrar etc.

Variables:

- pokemon: Este va a ser un map que contenga los datos de todos los pokemon registrados, utilizando como ID el nombre o ID del pokemon, y como su dato principal eltipo de dato Pokemon*, almacenando así punteros hacia objetos de pokemon.

Funciones:

- carga_pokemon(): Mandará a llamar al filemanager, el cual, se va a encargar de cargar todos los datos de pokemon de un archivo, para meter todos esos pokemon dentro de nuestro respectivo map.
- Guarda_pokemon(): Mandará a llamar una función al filemanager, el cual, se va a encargar de guardar todos los datos de nuestro map de pokemon hacia un archivo.

- Datos_pokemon(): Este deberá de regresar un objeto de pokemon dentro de nuestro map, utilizando ud id que recibimos como parámetro para usarlo como clave para encontrar dicho pokemon dentro de nuestro map, en caso de encontrarse, regresar el puntero hacia ese objeto de pokemon encontrado, en caso de que no se haya encontrado, regresar un NULL.
- Obtiene_todos_pokemon(): Esta función deberá de regresar el map que utilizamos de registro para todos los pokemon que se tengan.
- Nuevo_pokemon(): toma todos los datos que le pase la ventana, en su defecto, un Pokemon* para poder almacenarlo dentro del map que contiene los datos de todos los pokemon, entonces almacenarlo de ser posible dentro de un archivo.
- Busca_pokemon(): toma como parámetro el id que manejes para administrar los pokemon, entonces busca en el map si se encuentra el dato, en caso de que si, regresar el Pokemon*, en caso de que no, regresar NULL para indicar de que no se encontró.
- Filtra_pokemon(): Toma como parámetro un ID de un tipo, entonces, regresa una lista con todos los Pokemon* que coincidan con el tipo(s) seleccionado(s).

Clase Pokemon: Contiene todos los atributos necesarios para manejar toda la información de cada uno de los pokemon tal cual como aparece en el diagrama.

Atributos

- Id: Almacenará un número que no se repite que será el identificador del pokemon.
- Nombre: Almacenará el nombre del pokemon, estos tampoco se deberán repetir.
- Evolución: tiene el ID de una posible evolución.
- Preevolución: tiene el ID de una posible preevolución.
- Descripción: Contendrá alguna descripción para el pokemon.
- Tipo 1: Contendrá el tipo del pokemon.
- Tipo 2: Contendrá un posible tipo secundario del pokemon.
- imgPath: Tendrá la ruta hacia una imagen que represente visualmente al pokemon.

Clase FileManager: Una clase que tiene funciones para administrar los archivos y sus datos que conlleva el programa.

Atributos

- pokemonListPath: Contiene una cadena con la ruta en la que se almacenará el archivo con los datos.

Funciones

- Save_pokemon(): Toma el map de pokemon para que sea iterado y almacenado dentro de un archivo json.
- Load_pokemon(): Toma un archivo .json y carga todos los datos de los pokemon dentro de un map con los datos respectivos.

Clase NewPokemonWindow: Es una ventana que hereda de QDialog, la cual sirve para obtener los datos de un pokemon y almacenarlos dentro de un objeto Pokemon, a su vez, también puede recibir un objeto de pokemon para que este pueda ser modificado en sus datos.

Atributos

- Modo: almacenará el tipo de modo en que se estará usando la ventana, nuevo o editar.

Funciones:

- Toma_datos(): Se dedica a obtener todos los datos de la interfaz y los guarda dentro de un objeto de Pokemon*.
- Toma_info_pokemon(): Regresa un Pokemon* con todo el contenido que se ingresó en la interfaz.

Requerimientos:

- Se tendrá un área para mostrar la imagen de algún pokemon.
- Se tendrá un área para mostrar los datos del pokemon.
- Se tendrá un área para widgets para poder buscar y filtrar pokemon.
- Se tendrá un área para mostrar la lista de pokemon disponibles.
- Para mostrar los datos, en la parte de evoluciones y preevoluciones, si se tienen alguna, deberás de mostrar el nombre correspondiente del pokemon referenciado, no el ID.
- El tipo del pokemon lo mostrarás con su respectivo texto.
- Los tipos que podrán tener los pokemon son los siguientes: agua, fuego, planta, eléctrico, tierra, roca, volador, veneno, normal, lucha, psíquico, oscuridad, fantasma, dragón, hielo, acero, hada, bicho.
- El manejo de los archivos .json para almacenar los pokemon, deberá de ser transparente.
- Cuando registres un nuevo pokemon, tendrás que copiar el archivo de imagen del cual se seleccionó hacia una carpeta en la raíz del proyecto llamada “img_pokemon”.
- Cuando le agregas un tipo a algún pokemon dentro de la ventana de nuevo pokemon, este tiene que ser manejado con un QComboBox, evitando que se agregue directamente un ID como texto.

Entregables:

- Se deberá entregar todo dentro de una carpeta con el mismo formato del nombre que se ha manejado durante el curso.
- Dentro de esa carpeta habrá una carpeta llamada “src” que únicamente contendrá el código del proyecto.
- Dentro de la carpeta del proyecto, habrá una carpeta llamada “documentos” donde vendrá su reporte, el cual tendrá también el mockup/esquema para armar la pokedex, indicando como acomodaron los widgets.
- Dentro de la carpeta del proyecto, habrá una carpeta llamada “deploy” donde vendrá un ejecutable de su programa en forma de Release/deploy listo para usar dándole doble clic o enter.
- Si se encuentra un archivo .ui queda descalificado y se calificará con 0.