1. Introducción a Interfaces

Las clases, sus instancias (los objetos) y el acceso a los objetos usando variables referenciadas son la base de la programación orientada a objetos en ABAP. Hay veces en las que es necesario para clases similares proporcionar funcionalidades similares, pero que están codificadas diferentes en cada clase.

1. ¿Qué son las Interfaces?

Son estructuras independientes que se pueden implementar en una clase para extender el ámbito de esa clase.

El ámbito específico de una clase viene definido por sus componentes y sus secciones de visibilidad. Los componentes protegidos de una clase definen su ámbito en lo que se refiere a sus subclases. Las interfaces extienden el ámbito de una clase añadiendo sus propios componentes a la sección pública. Esto permite a los usuarios acceder a diferentes clases por medio de un punto de contacto común

Las interfaces junto con la herencia proporcionan uno de los pilares básicos del polimorfismo, ya que permiten que un sólo método con una interface se comporte distinto en diferentes clases

Las interfaces permiten usar diferentes clases de una manera uniforme aprovechando las referencias a las interfaces, es decir, el polimorfismo, por ejemplo las interfaces implementadas en diferentes clases amplían el ámbito público de la clase en el mismo conjunto de componentes. Si la clase no tienen ningún componente público específico de ella misma entonces la interface describe completamente el ámbito público de la clase.

1. Definición de una Interface

Las interfaces se pueden definir o bien globalmente en el Repositorio R/3 o localmente en un programa ABAP. **La definición de una interface** local es el código existente entre las sentencias siguientes:

**INTERFACE <INTERFACE>.**

**...**

**ENDINTERFACE.**

La definición contiene **la declaración de todos los componentes (atributos, métodos y eventos)** de la interface. Se pueden definir los mismos componentes en una interface que en una clase

No tienen que ser asignados a ninguna sección de visibilidad, ya que automáticamente pertenecen a la sección pública de la clase. Las interfaces no tienen una parte de implementación, ya que en sus métodos son implementados en la clase que implementa la interface.

1. Implementación de Interfaces

Cuando se implementa una interface en una clase, los componentes de la interface se añaden al resto de componentes de la sección pública. Un componente de una interface puede ser direccionado como si fuese un miembro de la clase bajo el siguiente nombre:

**<INTERFACE-COMPONENTE>.**

La clase tiene que implementar los métodos de todas las interfaces implementadas en ella. La parte de implementación de la clase debe contener la **implementación de cada método** de la interface:

**METHOD <INTERFACE-METODO>.**

**...**

**ENDMETHOD.**

1. Referencias a Interfaces

Las variables referenciadas permiten acceder a los objetos. En lugar de crear variables referenciadas con referencia a una clase, se pueden crear con referencia a una interface. Para **definir una referencia a una interface** se usa la siguiente adición: **TYPE REF TO <INTERFACE>**

La interface que utilicemos debe haber sido declarada en el programa.

Una variable referenciada con el tipo referenciado a una interface se llama variable referenciada a una interface, o referencia a interface simplemente

Una referencia a interface permite al usuario usar la siguiente declaración: **<IREFERENCIA>-><COMPONENTE>**

1. Direccionar y asignar objetos usando referencias a interfaces

Primero debemos declarar una variable de referencia a la clase, para poder crear un objeto de la clase.

Si la clase implementa una interface, se puede usar la siguiente asignación entre la variable referenciada a la clase y una referencia a interface, para hacer que la referencia a interface apunte al mismo objeto que la referencia a clase: **<IREFERENCIA> = <CREFERENCIA>**

Usando la **variable referenciada a una clase**:

* Para acceder a un **atributo** utilizaremos lo siguiente:

**<CREFERENCIA>-><INTERFACE-ATRIBUTO>**

* Para acceder al **método** utilizaremos lo siguiente:

**CALL METHOD <CREFERENCIA>-><INTERFACE-METODO>**

Usando la **variable referenciada a una interface**:

* Para acceder a un **atributo** utilizaremos lo siguiente:

**<IREFERENCIA>-><ATRIBUTO>**

* Para acceder al **método** utilizaremos lo siguiente:

C**ALL METHOD <IREFERENCIA>-><METODO>**

Siempre que los **componentes estáticos de las interfaces estén implicados**, solo se puede usar **el nombre** de la interface para acceder a las constantes:

* Para acceder a una constante utilizaremos lo siguiente:

**<INTERFACE>=><CONSTANTE>**

Para todos los demás componentes estáticos de una interface, solo se pueden usar **referencias** a objetos o la clase que implementa la interface:

* Para acceder a un atributo estático utilizaremos lo siguiente:

**<CLASE>=><INTERFACE-ATRIBUTO>**

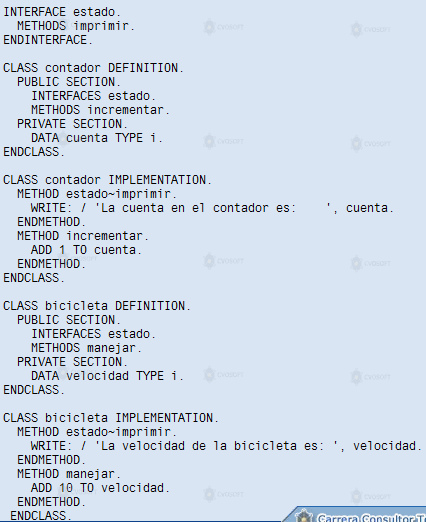
* Para llamar a un método estático utilizaremos lo siguiente:

**CALL METHOD<CLASS>=><INTF-METH>**

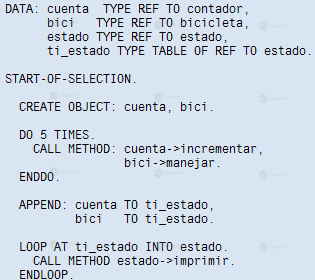
1. Ejemplo de Interfaces

Vemos un ejemplo de como implementar las interfaces con sus respectivas declaraciones.

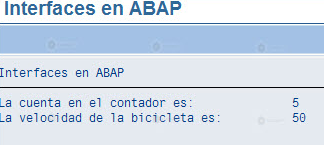
Se declaran la interface ESTADO y las clases CONTADOR y BICICLETA. Dentro de la sección pública de ambas clases declaramos la interface ESTADO.



Luego dentro del evento START-OF-SELECTION creamos los objetos CUENTA y BICI. Ejecutamos los métodos CUENTA->INCREMENTAR y BICI->MANEJAR, almacenamos los resultados en la tabla TI\_ESTADO y recorremos la tabla llamando al método IMPRIMIR.



Salida por pantalla:



Las interfaces son tipos de objetos que se pueden definir localmente en un programa ABAP o bien globalmente a través del repositorio R3 en la transacción SE24.

Si se definen interfaces **globalmente** entonces estas son almacenadas en un tipo especial de programa ABAP llamado **INTERFACE POOL** el cual es del tipo j y sirve como contenedor del respectivo tipo de objeto. **Cada pool de interfaces contiene la definición de una sola interface** este programa es generado automáticamente cuando creamos la interface global. Un pool de interfaces es comparable a un module pool o aun grupo de funciones. Los pool de interfaces no contienen ninguna sentencia ejecutable, cuando se implementa una interface en una clase, la definición de la interface queda implícitamente incluida en la definición de la clase.