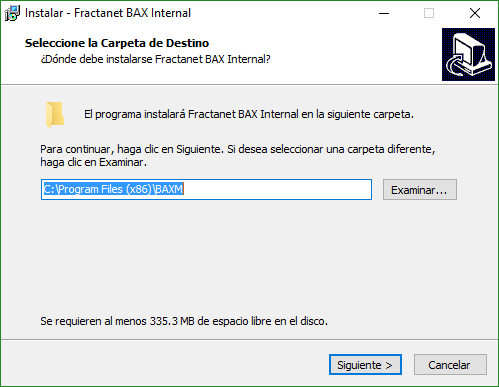
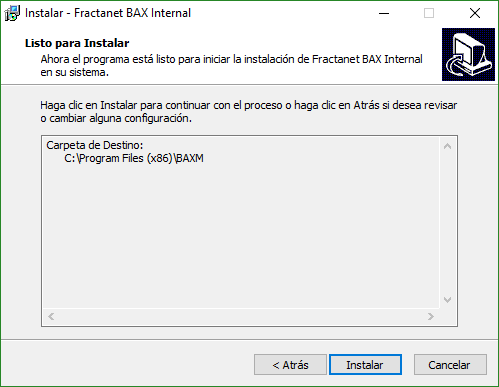
# BAX 7.5

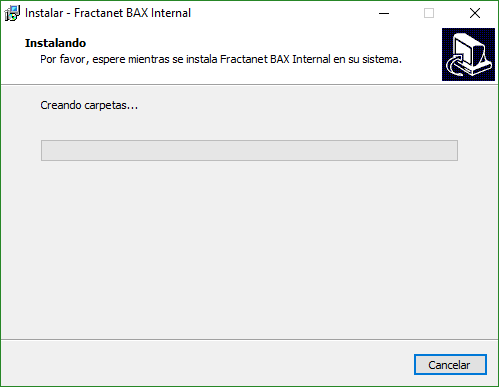
Es una herramienta de diseño y generación de aplicaciones web basada en la metodología RAD, que permite tener aplicaciones sencillas en funcionamiento en muy poco tiempo, pudiendo concentrar el esfuerzo en la resolución de las reglas de negocio para así reducir los tiempos de entrega.

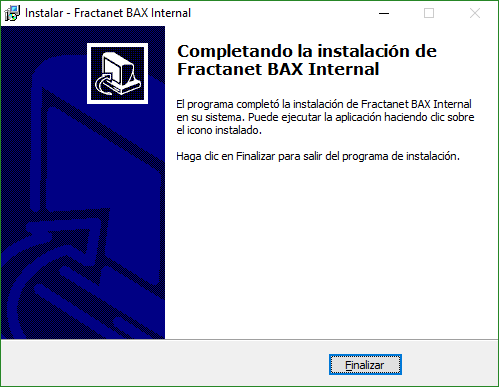
## Instalación

La instalación de BAX es muy sencilla ya que se tiene el apoyo de un instalador muy sencillo solo es necesario seguir los pasos y se instalara automáticamente.



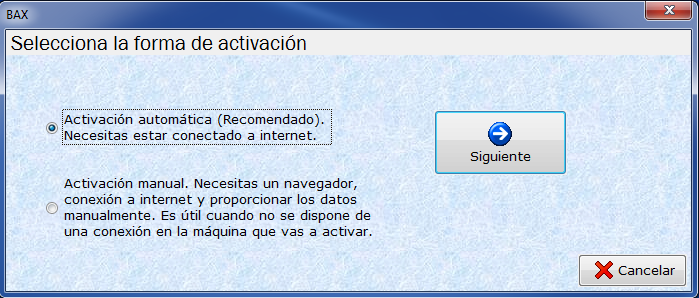






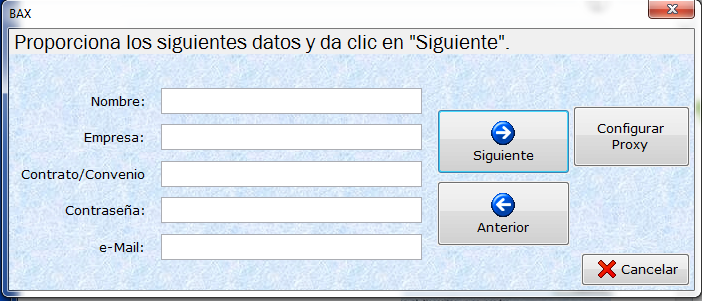
Ya instalado BAX y cuando se corre por primera vez la aplicación nos aparecerá el dialogo de activación, a continuación, se mostrará como activar una cuenta de manera automática.

Primero seleccionaremos Activación automática como se muestra a continuación.

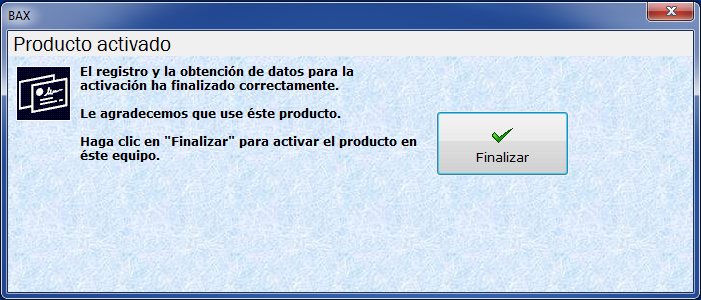


Posteriormente nos pedirán los siguientes datos:

* Nombre: Nombre de la persona que tendrá la licencia.
* Empresa: empresa para la cual labora.
* Contrato/Convenio: El nombre del convenio o el contrato (este dato lo proporciona FRACTANET).
* Contraseña: Contraseña brindada por FRACTANET.
* E-Mail: Correo electrónico de la persona que posera la licencia.



Si los datos ingresados son correctos nos aparecerá la siguiente ventana indicándonos que el producto ha sido activado, solo damos clic en el botón de finalizar y nos abrirá BAX para empezar a trabajar.



## Crear un proyecto

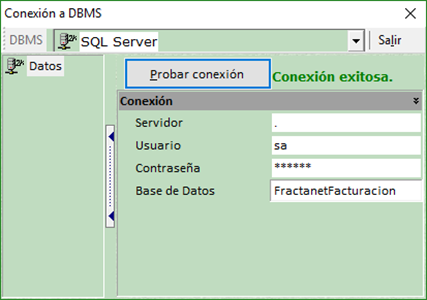
Para comenzar un proyecto solo es necesario iniciar la aplicación y aparecerá la siguiente pantalla.



En ella nos piden los siguientes datos:

* Nombre del proyecto: El nombre que se le dará a todo el proyecto, incluyendo a al modelo.
* Autor: El nombre de usuario que tiene la licencia de BAX.
* Nombre de la aplicación: Nombre que le daremos a la aplicación.
* Nombre del ejecutable: Nombre que se le dará al archivo ejecutable.
* Para la empresa: Nombre de la empresa a la que se creara la aplicación.
* Versión: Numero de la versión en la que iniciara el proyecto.
* Guardar en: La ruta en la cual se guardarán el modelo y la aplicación.

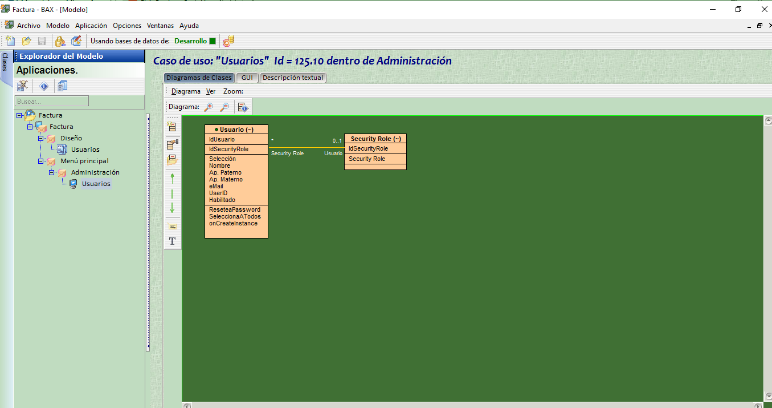
Ya con los campos llenos oprimimos el botón de aceptar y nos aparecerá la siguiente ventana, en la cual nos piden los datos que se ocuparan para la conexión hacia la Base de Datos.



En ella pondremos el nombre del servidor, un usuario y su contraseña para acceder a la base de datos, y el nombre que se le dará a la base de datos que ocuparemos en el proyecto.

Ya con los datos establecidos damos clic en el botón de Probar conexión, si todos los datos están correctos nos aparecerá la leyenda de conexión exitosa como se ve en la imagen anterior.

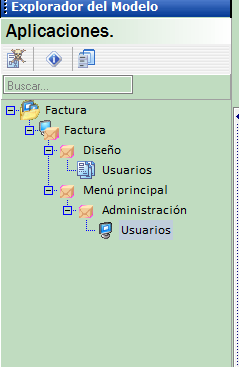
Solo resta dar clic en el botón de salir y se creará el proyecto, de igual manera abrirá el modelo que genera BAX para la creación de la aplicación como lo vemos a continuación.



## Modelar el menú de la aplicación

Empecemos a modelar el menú principal de nuestra aplicación en BAX.

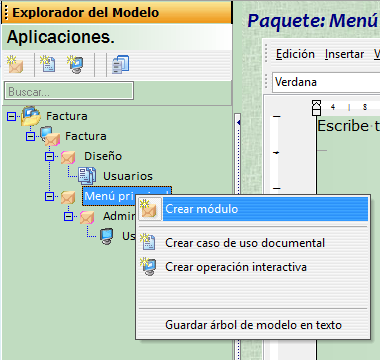
Ya con la pantalla principal de nuestro proyecto iremos al explorador del modelo, que se encuentra en la parte izquierda de la ventana.



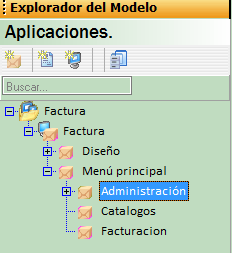
Este explorador se divide en dos partes principales, la primera es el área de diseño y nos ayuda para modelar y diseñar la base de datos, y la segunda es el área del menú de la aplicación, aquí crearemos el menú de la aplicación.

Antes de crear el menú es importante definir que es un módulo. Un módulo es la representación en el modelo de BAX de un sub-menú, esto quiere decir que cuando generemos la aplicación los módulos que se encuentren dentro del área de menú principal se convertirán en un sub-menú, los módulos se pueden identificar con el siguiente símbolo.  .

Una vez definido que es un módulo y que es un caso de uso creemos el modelo de nuestra aplicación, el primer paso es ubicar el puntero sobre el módulo de menú principal y damos clic derecho, elegiremos a la opción de crear modulo como se ve en la imagen.



Una vez con el modulo creado le damos un nombre y queda como en la siguiente imagen.



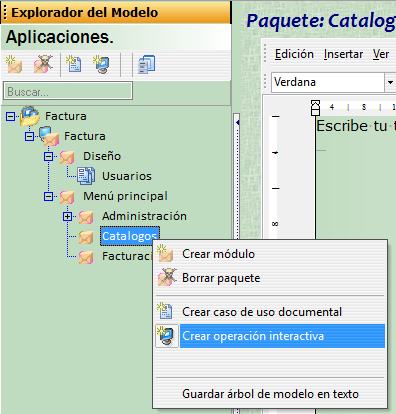
Así se crea el modelo del diseño del menú, si es necesario se pueden agregar más módulos de la misma manera.

## Modelar las pantallas de la aplicación

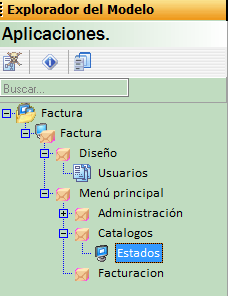
Para crear las pantallas que tendrá la aplicación se ocuparan casos de uso.

Un caso de uso es la representación en BAX de una pantalla o ventana de la aplicación, se identifican con este símbolo. 

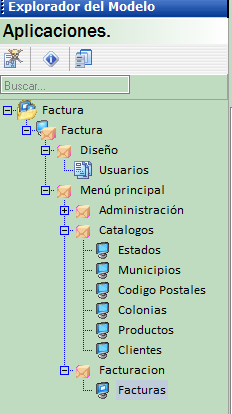
Veamos un ejemplo de cómo crear un caso de uso, para esto ponemos el cursor sobre el módulo de Catálogos, damos clic derecho y elegimos la opción de crear operación interactiva como se observa en la siguiente imagen.



Le ponemos el nombre con el cual queremos que se muestre en la aplicación y quedando de la siguiente manera.

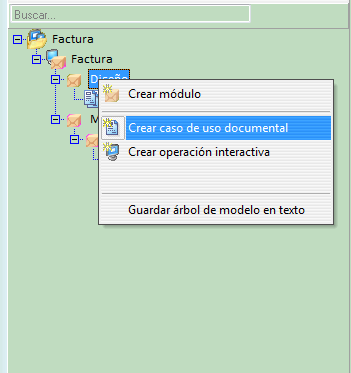


Podemos crear tantos casos de uso como sean necesarios

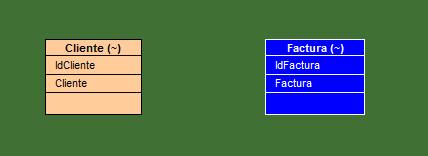


## Modelar la base de datos

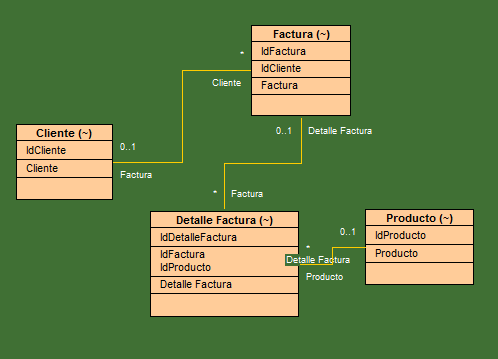
En la carpeta de diseño daremos clic derecho y escogeremos un caso de uso documental.



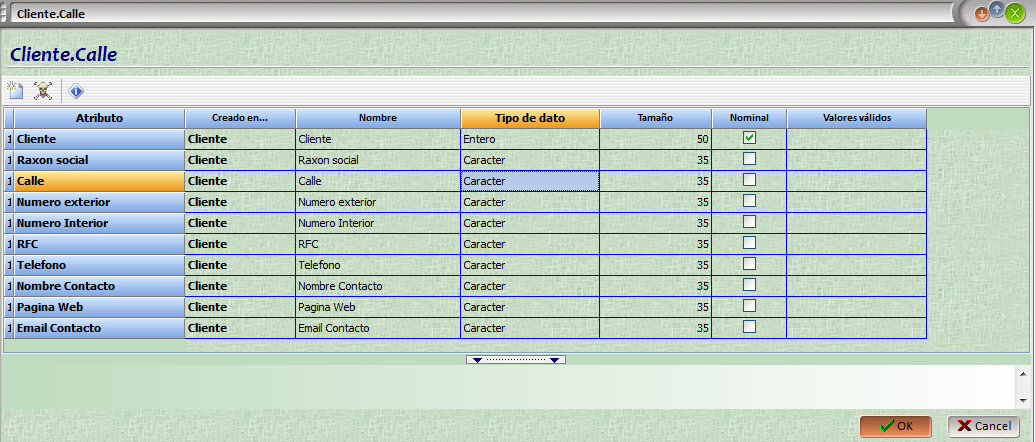
En este caso de uso crearemos el modelo de la base, para poner una nueva clase apretaremos este botón , este botón nos creara un caso de uso a este le pondremos el nombre de nuestra entidad que necesitamos.



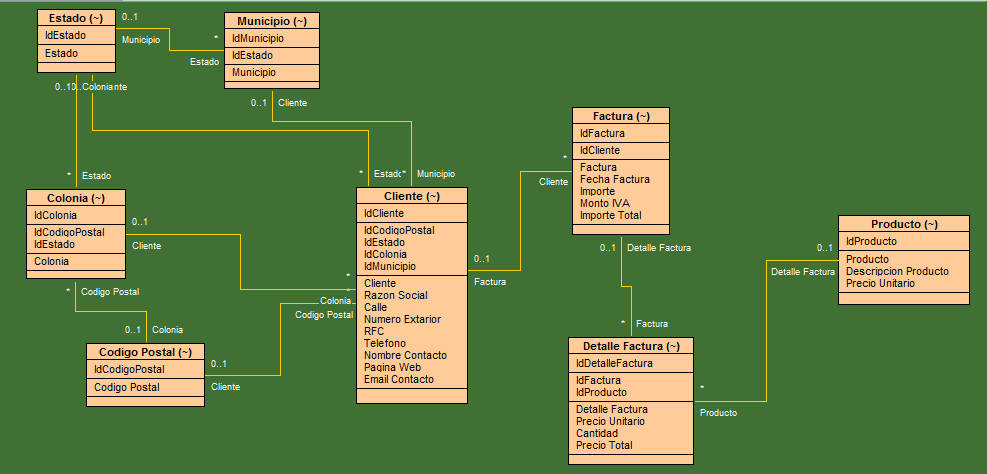
Las relaciones se generan con el siguiente botón , y se erigirá las dos clases la cual tendrán relación dándoles clic a cada una.



También se pueden agregar atributos a una clase, esto dando doble clic sobre la clase que se quiere poner más atributos, se abrirá una nueva ventana en donde se agregaran los atributos, se pondrá el nombre del atributo, el tipo de dato y el tamaño.

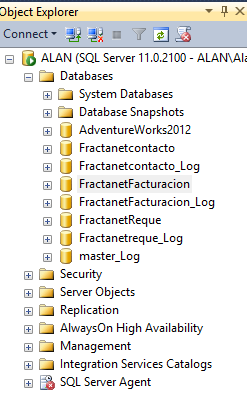


Lo mismo se hará con las demás clases que necesiten más atributos al terminar el modelo terminará de la siguiente manera.

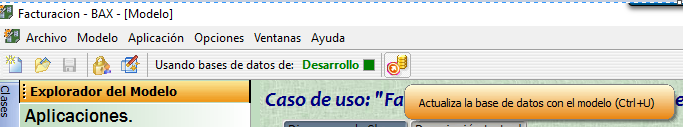


## Generar la base de datos

Como podemos ver en nuestro manejador de bases de datos actualmente no existe la base de datos que ocupara la aplicación en este caso sería FractanetFacturacion.



Una vez que se a terminado el modelado de la base de datos, iremos a al menú principal y buscaremos el botón de generar la base de datos como se ve en la figura.



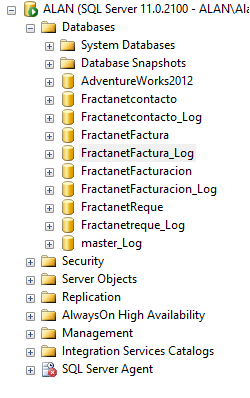
Damos clic y nos aparecerá la siguiente ventana.



Aquí oprimiremos el botón de agregar todas las tablas y posteriormente el botón de actualizar, esperaremos a que se realice la acción y nos mandara en la parte inferior un mensaje de acción finalizada.

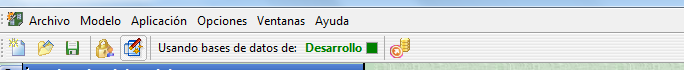


Como podremos observar en nuestro manejador de base de datos, se crearon dos bases una con el nombre de la empresa+nombre del proyecto (FractanetFactura) y otra es empresa+proyecto\_log, esta última contendrá un historial de los errores que presenta la aplicación, es muy importante tener en cuenta esto, ya que si queremos saber qué tipo de error es el que está presentando podemos acudir a esta base y verificarlo.

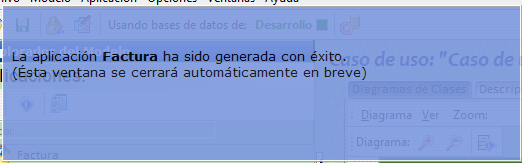


## Generar la aplicación

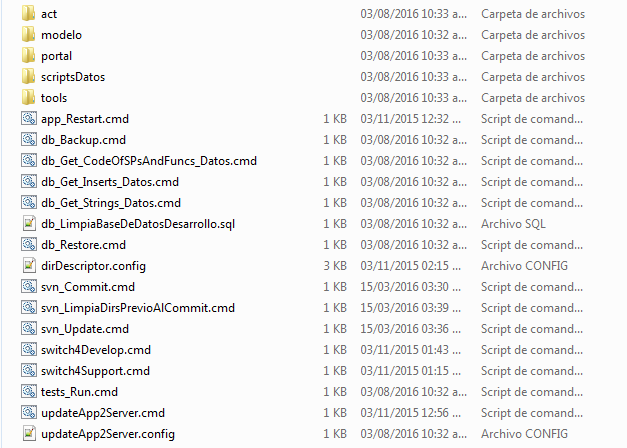
Una vez teniendo nuestro modelo y la creación del menú terminado se generará la aplicación para esto iremos al menú principal y buscamos el botón de "Generar la aplicación actual" y damos clic.



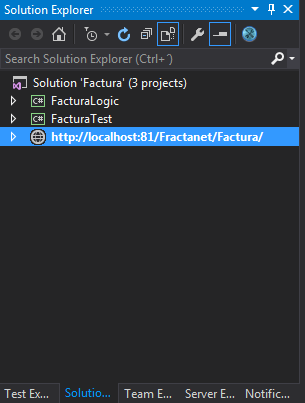
Se empezará a generar la aplicación al terminar mostrara un mensaje diciendo que la aplicación fue generada de color azul como el siguiente. Nota: la primera vez que se genere la aplicación puede tardar varios minutos.



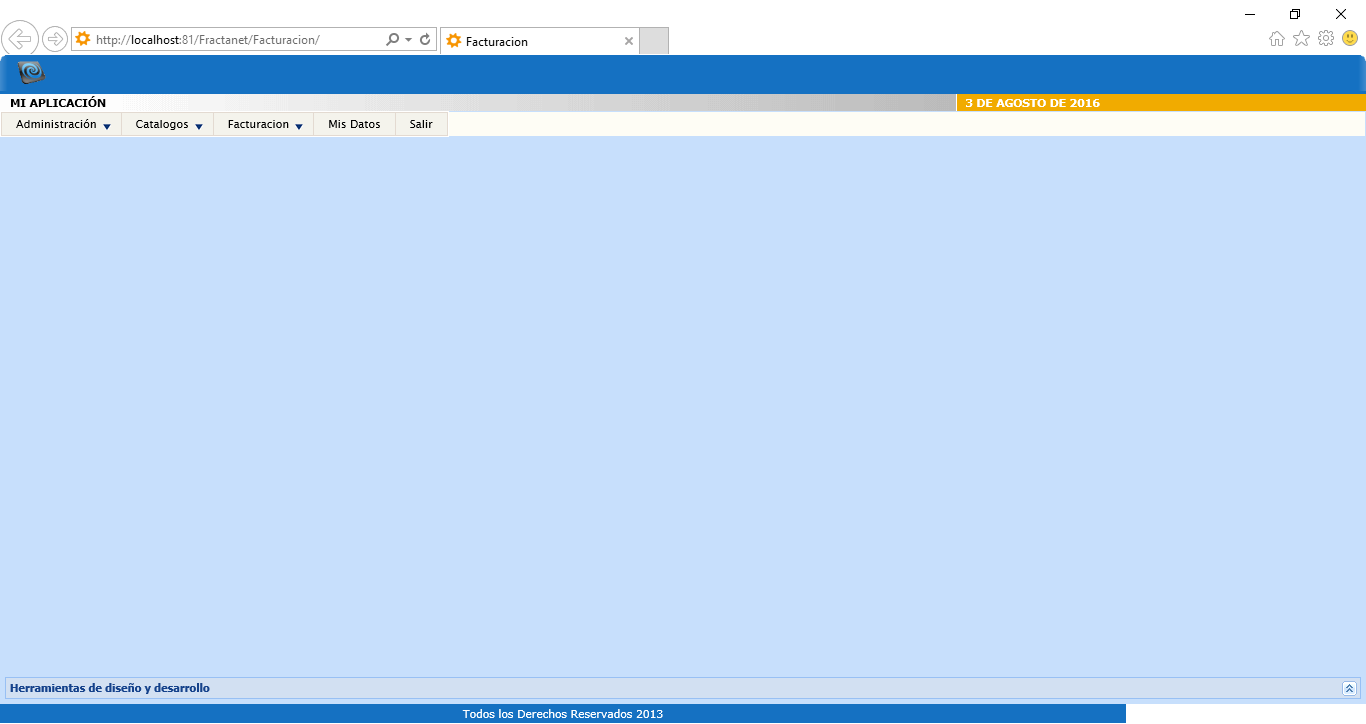
Se creará una carpeta en la ruta donde fue guardada la aplicación con el nombre de la empresa y adentra se encontrará otra carpeta con el nombre de la aplicación esta contendrá varias carpetas en las cuales contendrá la aplicación, el modelo y demás configuraciones al igual que archivos .bat que de igual manera nos ayudaran a realizar algunas configuraciones de la aplicación



La aplicación se encuentra en la carpeta de \act se abrirá el archivo con Visual Studio, la solución contendrá 3 proyectos de Logic, Test y la aplicación Web.



Podemos correr la aplicación con F5 y verla funcionar

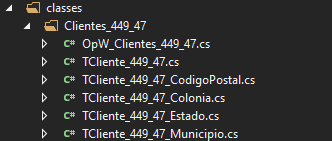
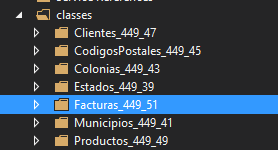


## Descripción del código fuente generado

La solución contiene 3 proyectos los cuales son:

* **Logic**: en ella se encuentra toda la lógica de negocio, clases del modelo, los scripts de SQL y clases auxiliares.
* **Test**: se encuentra la lógica de las unidades de prueba, en ella se realizaran las pruebas necesarias para comprobar que la aplicación haga lo que debe.
* **Aplicación**: es el punto de entrada y la implementación de GUI, las páginas web creadas a mano, plantillas, todo lo que es la parte de diseño se guarda aquí.

El proyecto de lógica va a contener las clases, estas clases van a contener el nombre de cada pantalla realizada en BAX anteriormente y con un número que va a ser el identificador de cada clase, dentro de cada una de ella va a contener un archivo OpW es la lógica de las pantallas, contiene todos los eventos que pueden hacer cada pantalla en diferentes situaciones insertar, actualizar, antes de guardar, al borrar, etc. , también contiene subclases y las relaciones a esas subclases así como se muestran en las siguientes imágenes.



Dentro de la aplicación encontramos dos directorios importantes:

* **App\App\_Code**: se encuentra código APS.NET y se utiliza para crear eventos dentro de la app
* **App\App\_Data**: en esta parte se guardan múltiples configuraciones de la app el diseño, orden de las columnas, código javascript, plantillas, etc.

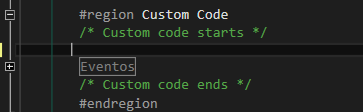
## Escribir código

La mayor facilidad que nos da BAX es poder tener una aplicación de una manera muy fácil y rápida, permitiéndonos enfocarnos más en las reglas del negocio, en este punto es cuando la codificación entra en escena.

Como vimos anteriormente BAX genera clases y opw y estas contienen métodos que ocuparemos para introducir las reglas de negocio.

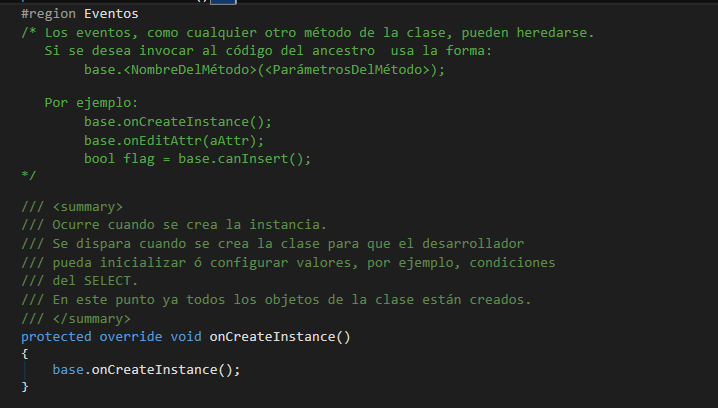
El primer paso es encontrar el modulo de "Custom Code" ya sea en las clases o en las opw.

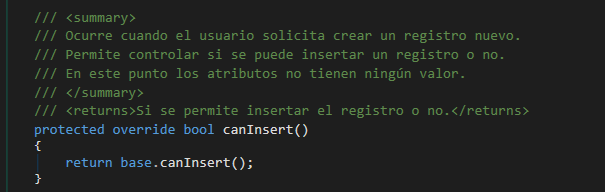


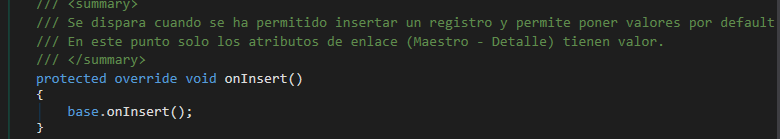


Es importante que todo el código que se vaya a generar sea puesto dentro de este modulo, y que si se vuelve a genera la aplicación solo lo que este dentro de custom code será preservado.

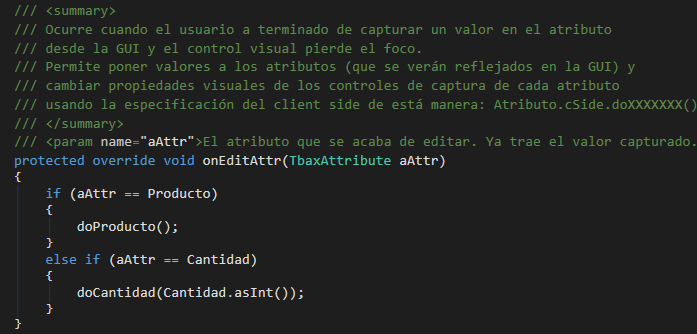
Dentro de custom code existe otra área llamada evento, aquí se encuentran una serie de eventos que pueden suceder cuando se estén ocupando las clases, cada evento tiene una breve explicación de su funcionamiento y cuando entra en acción.





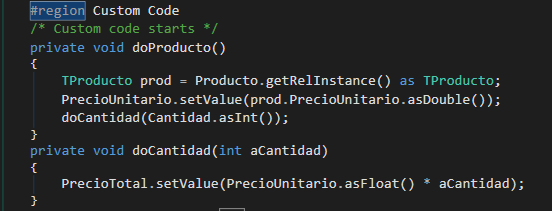


En estos eventos es donde nosotros pondremos nuestro código a continuación veremos un ejemplo.



En este ejemplo ocupamos el evento onEditAttr el cual nos indica que se aplica cuando capturamos un valor en un atributo definido y después ese atributo pierde el foco de atención, es decir cuando editamos un atributo de la interfaz y después elegimos otro, o damos clic a cualquier otro lado de la pantalla.

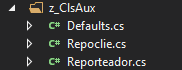
En este ejemplo cuando el campo editado sea el de Producto, se ejecutará un método llamado doProducto y si es cantidad se ejecutará el método de doCantida, podriamos poner directamente en esta zona los métodos antes mencionados, pero estaríamos violando uno de nuestros principios de arquitectura, y ese seria el principio de LOC, que nos indica que nuestra programación o nuestro número de líneas cuando programamos debe tender a CERO. Debido a esto estos métodos los separaremos y los pondremos en el área de custom code como se puede ver en la siguiente imagen.



Estos métodos también podrían ser separados y puestos en una clase diferente, con la finalidad de poder pedir ayuda, supongamos que un compañero tiene disponibilidad para ayudarnos a crear estos métodos, lo único que necesitara hacer es crear una nueva clase en la carpeta de z\_ClsAux.



En esta carpeta podemos crear clases auxiliares que necesitemos y no serán borradas en nuevas generaciones de la aplicación.



Como se puede observar la codificación en una aplicación generada con BAX también se simplifica, al solo necesitar saber cuándo se tiene que realizar la acción que deseemos.