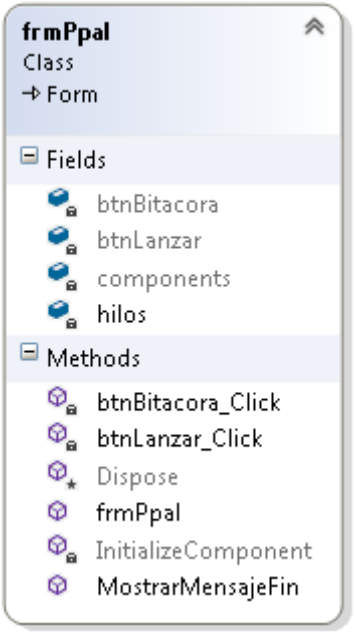
Universidad Tecnológica Nacional Facultad Regional Avellaneda										
Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos										
Materia: Laboratorio de Programación II										
Apellido:				Fecha:		22/06/2017				
Nombre:				Docente(2):		Federico Dávila				
División:	2ºE			Nota(2):						
Legajo:				Firma(2):						
Instancia(1):	**PP**		**RPP**		**SP**	X	**RSP**		**FIN**	

(1) Las instancias validas son: 1er Parcial (**PP**), Recuperatorio 1er Parcial (**RPP**), 2do Parcial (**SP**), Recuperatorio 2do Parcial (**RSP**), Final (**FIN**). Marque con una cruz.
(2) Campos a ser completados por el docente.

- El nombre de la carpeta del proyecto debe tener el formato: apellido.nombre.división. Por ejemplo Pérez.Juan.2E.
- **Los proyectos que no sean identificables, no serán corregidos.**
- **Los alumnos que no entreguen o su parcial no sea identificable serán desaprobados.**
- **El proyecto que no COMPILE será desaprobado.**
- Sólo se corregirá lo que el alumno entregue de la siguiente forma:
  - Al finalizar, colocar la carpeta de la Solución completa en un archivo ZIP y dejar este último en el Escritorio de la máquina. Luego presionar el botón de la barra superior, cargar un mensaje y presionar Aceptar. La barra superior deberá cambiar de color.
- En todos los casos que sea posible, rehutilizar código.

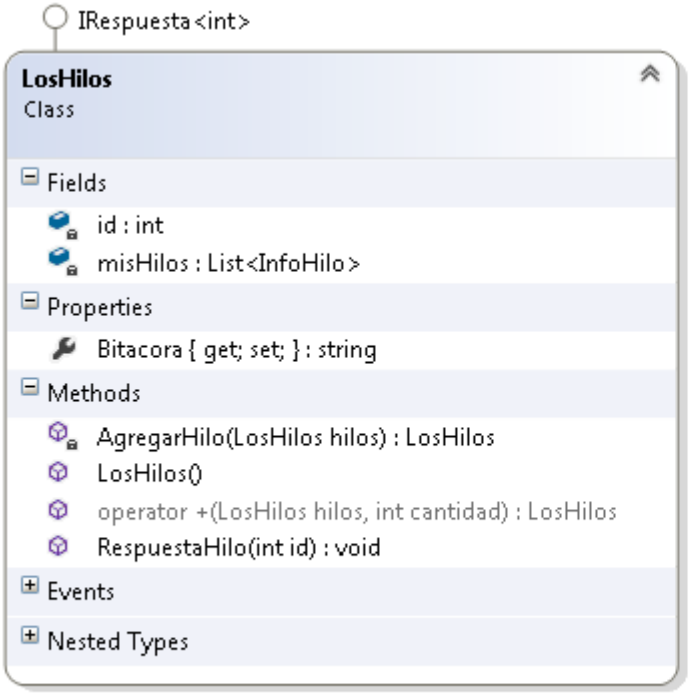
1. Generar un proyecto con el nombre del alumno del tipo Windows Form. El formulario deberá verse exactamente como este:

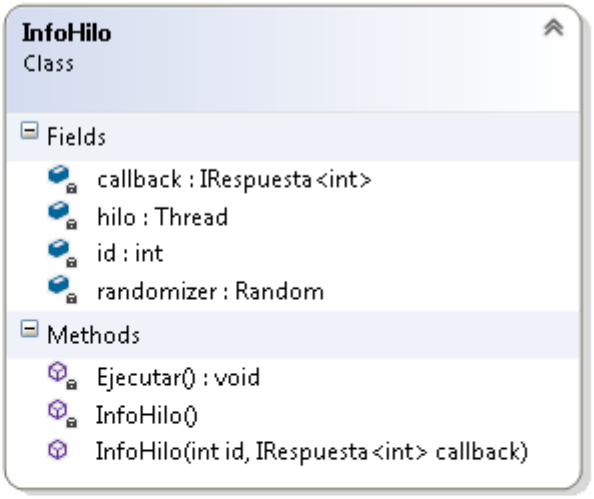
2. Y la clase del formulario tendrá el siguiente formato:



3. Generar un proyecto llamado Interfaces y dentro colocar la siguiente interfaz:

4. Generar un proyecto llamado Entidades y dentro colocar las siguientes clases:





5. Formulario:
   a. Crearemos el atributo *hilos*.
   b. El evento *AvisoFin* de *hilos* deberá estar relacionado con el método *MostrarMensajeFin* del formulario.
   c. *MostrarMensajeFin* mostrará por pantalla el mensaje recibido.

    d. Al presionar el botón Lanzar se deberá, mediante la sobrecarga del +, agregar un nuevo hilo al atributo *hilos*. En caso de error, se mostrará mediante un MessageBox.
    e. Al presionar el botón Bitacora se deberá mostrar por pantalla la bitacora guardada por *hilos*.
6. La interface IRespuesta sólo contará con el método *RespuestaHilo* que recibirá un atributo genérico *id*.
7. Clase InfoHilos:
    a. Atributo estático *randomizer*. El mismo se inicializará en un constructor de clase.
    b. El Thread *hilo* ejecutará el método *Ejecutar*.
    c. Ejecutar frenará el código durante un tiempo aleatorio de entre 1 y 5 segundos. Luego de transcurrir este tiempo, utilizará el método *RespuestaHilo* de *callback* pasando como parámetro el atributo *id*.
8. Clase LosHilos:
    a. El atributo id deberá inicializar el 0.
    b. La propiedad Bitacora utilizará el set para generar un archivo en el escritorio de la máquina donde se ejecute llamado "bitacora.txt". El get retornará el contenido del mismo archivo.
    c. Método de clase AgregarHilo hará los siguientes pasos, en el siguiente orden:
        i. Incrementará id.
        ii. creará un nuevo InfoHilo y lo agregará a *misHilos*.
    d. RespuestaHilo hará los siguientes pasos, en el siguiente orden:
        i. Creará un mensaje con el siguiente formato: `"Terminó el hilo {0}."`.
        ii. Guardará el mensaje en la bitácora.
        iii. Ejecutará el evento AvisoFin.
    e. El operador + lanzará la excepción CantidadInvalidaException en el caso de que la cantidad sea menor a 1.
    f. Si cantidad es mayor a 0, deberá agregar tantos hilos cómo indique dicha cantidad.