|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Universidad Tecnológica Nacional**  **Facultad Regional Avellaneda** | | | | | | | | | | | | |
| Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos | | | | | | | | | | | | |
| Materia: Laboratorio de Programación II | | | | | | | | | | | | |
| Apellido: |  | | | | Fecha: | | | | 22/11/2018 | | | |
| Nombre: |  | | | | Docente(2): | | | | F. Dávila / H. Dillon | | | |
| División: | 2ºC | | | | 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.

1. ***IMPORTANTE:***

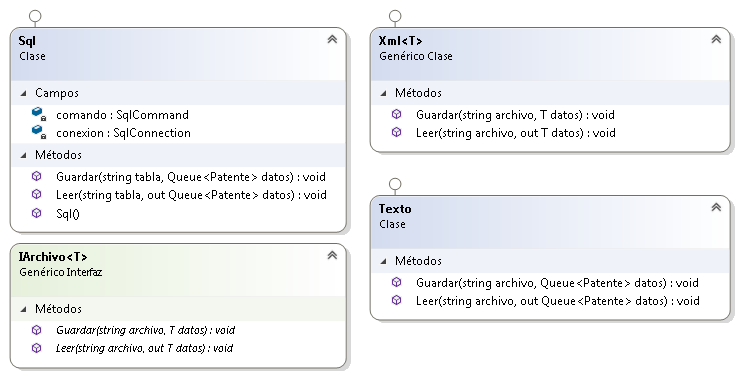
* ***2 (dos) errores en el mismo tema anulan su puntaje.***
* **La correcta documentación y reglas de estilo de la cátedra serán evaluadas.**
* Colocar sus datos personales en el nombre del proyecto principal, colocando: Apellido.Nombre.Departamento. Ej: Pérez.Juan.2D. No sé corregirán proyectos que no sea identificable su autor.
* **TODAS** las clases deberán ir en una Biblioteca de Clases llamada Entidades.
* No se corregirán exámenes que no compilen.
* **Reutilizar** tanto código como crean necesario.
* Colocar nombre de la clase (en estáticos), **this** o **base** en todos los casos que corresponda.

*TIEMPO MÁXIMO PARA RESOLVER EL EXAMEN* ***90*** *MINUTOS.*

1. Partir de la solución entregada. Modificar su nombre con el siguiente formato: [APELLIDO].[NOMBRE]

Archivos

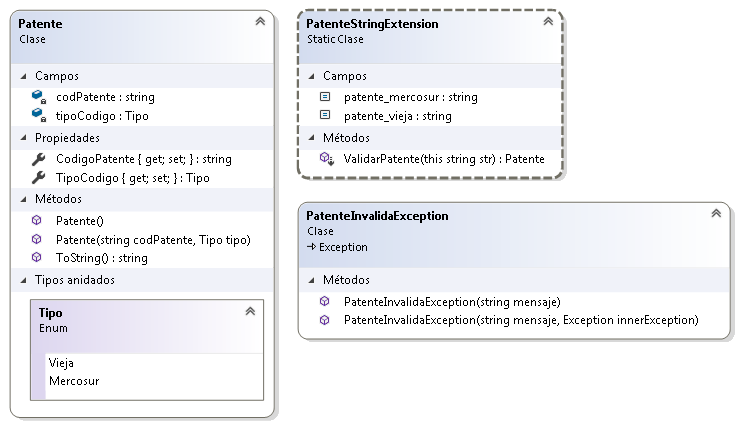
1. Dentro del proyecto **Archivos** se deberá respetar el siguiente esquema:



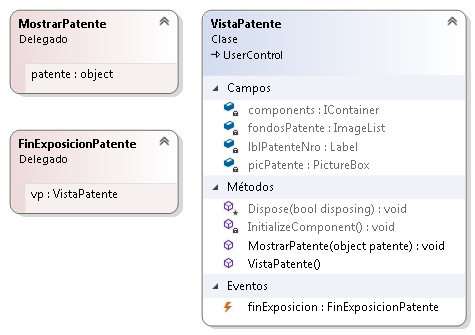
1. Las clases Sql, Xml y Texto implementarán la interfaz IArchivo para acceder y consultar los datos de una base de datos, un archivo serializado en formato Xml y un archivo de texto (provistos junto al examen y ubicados en el escritorio).
2. Sql recibirá el nombre de la tabla a consultar (patentes) y una cola con los datos.
3. Xml y Texto recibirán el nombre del archivo (**sólo el nombre**) y los datos a leer o guardar.

Entidades

1. Dentro del proyecto **Entidades** se deberá respetar el siguiente esquema:



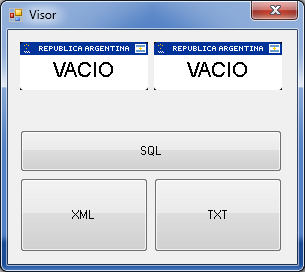
1. Dentro de la clase PatenteStringExtension se deberá colocar un método de extensión llamado ValidarPatente.
   1. ValidarPatente recibirá un string, y validará que coincida con los formatos de patente\_mercosur o patente\_vieja.
   2. Si es patente\_mercosur se deberá generar una nueva patente con tipo Mercosur.
   3. Si es patente\_vieja se deberá generar una nueva patente con tipo Vieja.
   4. Caso contrario se lanzará una excepción del tipo PatenteInvalidaException con el mensaje "{0} no cumple el formato."
2. Dentro del proyecto VistaPatentes agregar al control de usuario la siguiente funcionalidad:



1. Agregar dos delegados al NameSpace (**por fuera de la clase**) con el siguiente formato void FinExposicionPatente(VistaPatente vp) y void MostrarPatente(object patente).
2. Dentro del método MostrarPatente se deberá lograr que se muestre la patente durante un tiempo X y luego notificar por medio de un evento que finalizó dicha exposición. Completar debajo de los comentarios a fin de lograr la funcionalidad deseada.

Formulario

1. El formulario contará con 3 botones y dos controles de usuario. Al presionar los botones, se leerá de distintos orígenes una colección de patentes y se mostrarán en los controles de usuario:



1. Declarar un atributo del tipo lista de Threads. Inicializarlo en el constructor.
2. En el evento Closing del formulario, asegurarse de que todos los hilos estén terminados. Agregar el método FinalizarSimulacion que cumpla esa función.
3. Agregar el manejador del evento de la clase VistaPatente con el nombre ProximaPatente.
4. ProximaPatente, si hay elementos en la cola de patentes:
   1. Instanciará un hilo parametrizado para el método MostrarPatente del objeto VistaPatente recibido.
   2. Inicializará el hilo recién creado con el próximo elemento de la cola.
   3. Agregará el hilo a la lista.
5. En el evento Load del formulario, asociar el evento VistaPatente para los objetos vistaPatente1 y vistaPatente2 con el manejador.
6. En cada botón leer del origen que corresponda (SQL, XML, TXT) agregar los datos a la cola de Patentes. También capturar las excepciones y llamar al método IniciarSimulacion.
7. IniciarSimulacion:
   1. Finalizará los hilos activos.
   2. Llamará al método ProximaPatente para cada uno de los objetos del tipo VistaPatente del formulario.

Test Unitarios

1. Agregar 3 test unitarios, uno para cada método Guardar de las clases Xml, Sql y Texto. Cada uno de estos test generará un nuevo archivo, luego lo leerá y verificará que el dato leído sea correcto.

Al finalizar, colocar la carpeta de la Solución completa en un archivo ZIP que deberá tener como nombre Apellido.Nombre.division.zip y dejar este último en el Escritorio de la máquina.

Luego presionar el botón  de la barra superior, colocar un mensaje y apretar **Aceptar.**

Finalmente retirarse del aula y aguardar por la corrección.