

# Prova 4

## Avvertenze per la consegna

All'inizio di ogni file sorgente inserire in un commento i propri dati: **cognome, nome e numero di matricola**. Al termine dell'esame, **consegnare (inviare) tutti i file sorgenti**.

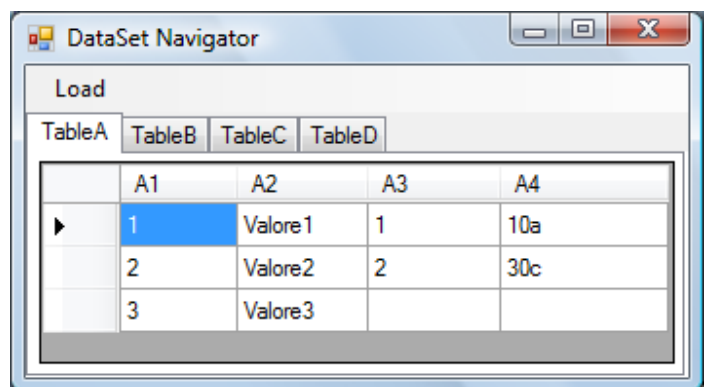
## Premessa

Si intende realizzare un componente in grado sia di costruire un **DataSet** leggendo da file il suo schema (in termini di strutture di tabella e relative relazioni), sia di popolare un **DataSet** già costruito leggendo da file una serie di dati congruenti con lo schema del **DataSet**.

## Passo 1 – Interfaccia di test

Creare un nuovo progetto di tipo *Windows Forms Application*, di nome **Prova4** e rinominare il file e la classe **Form1** in **DataSetNavigator**. Scaricare il file "Prova4Start.zip" contenente i file XML con schema e dati e copiare tali file nel direttorio contenente i file sorgenti dell'applicazione.

La *form* **DataSetNavigator** contiene il riferimento a un **DataSet** e deve apparire come in figura. Il **TabControl** deve contenere tante **TabPage** quante sono le **DataTable** presenti nel **DataSet** e sarà inizialmente vuoto. Ogni **TabPage** deve mostrare il contenuto di una **DataTable** mediante una **DataGridView** (con **Dock a Fill**). Il menù **Load** deve contenere le voci **Schema** e **Data**. Il menù



**Schema** dovrà consentire il caricamento di un nuovo schema nel **DataSet** (mediante **DataSetBuilder**). Il menù **Data** dovrà consentire il caricamento di dati nel **DataSet** (mediante **DataSetLoader**) se e solo se il **DataSet** è già stato costruito (contiene almeno una **DataTable**).

## Passo 2 – DataSetBuilder

Il metodo **Build** della classe **DataSetBuilder** deve creare un nuovo **DataSet** e lo deve riempire di tabelle e relazioni in base allo schema contenuto nel file XML il cui nome viene passato come argomento (per il formato del documento xml, vedere il contenuto del file "Schema.xml").

**Inserimento delle tabelle:** per ogni elemento **Table** contenuto nell'elemento radice **Schema** si inserisca nel **DataSet** una nuova **DataTable** creata tramite il

metodo **CreateTable** (da definire); se nel **DataSet** esiste già una tabella con lo stesso nome, occorre sollevare un'eccezione poiché lo schema non è consistente.

**Inserimento delle relazioni:** occorre considerare gli elementi **Column** con attributo **parentTable**. In particolare, si esegua un secondo ciclo su tutti gli elementi **Table**; per ogni elemento **Table**, si recuperi dal **DataSet** la **DataTable** relativa (**child**); per ogni elemento

DataSetBuilder	
+ Build ( [in] fileName : string ) : DataSet	
- CreateTable ( [in] tableElement : XmlElement ) : DataTable	

## Prova 4

**Column** contenuto in **Table** con attributo **parentTable**, si recuperi dal **DataSet** la **DataTable** (**parent**) il cui nome è dato dal valore dell'attributo; infine, si aggiunga al **DataSet** una nuova **DataRelation** in cui la **DataColumn** **parent** è la chiave primaria della **DataTable** **parent** e la **DataColumn** **child** è la colonna identificata dall'elemento XML che ha l'attributo **parentTable**. Per semplicità, le *primary keys* e le *foreign keys* sono sempre composte da una sola colonna.

Il metodo **CreateTable** costruisce una nuova **DataTable** a partire dall'elemento xml **tableElement** che ne descrive la struttura. Il nome della **DataTable** è dato dal valore dell'attributo **name** di **tableElement**. Per ogni elemento **Column** contenuto in **tableElement**, si deve aggiungere alla **DataTable** una nuova **DataColumn** il cui nome è dato dal valore dell'attributo **name** e il cui tipo è identificato dal valore dell'attributo **type**; se il valore dell'attributo **primaryKey** è "true", si deve definire come chiave primaria la nuova colonna (proprietà **PrimaryKey** di **DataTable**).

### Passo 3 – DataSetLoader

Il metodo **LoadData** di **DataSetLoader** deve caricare in un **DataSet** già costruito (parametro **dataSet**) i dati contenuti nel file XML il cui nome viene passato come secondo argomento (per il formato del documento xml, vedere il contenuto del file "Data.xml") –

verificare che entrambi i riferimenti non siano nulli, che **dataSet** contenga almeno una tabella e che **fileName** non sia una stringa vuota.

Al fine di effettuare il caricamento,

per ogni elemento **Table** contenuto nella radice **Data** del documento XML recuperare nel **DataSet** tramite l'attributo **name** la **DataTable** relativa (se non esiste una **DataTable** con quel nome sollevare un'eccezione); per ogni elemento **Row** contenuto nell'elemento **Table** aggiungere una nuova **DataRow** alla tabella tramite il metodo **AddRow**.

Il metodo **AddRow**, dati una **DataTable** (**table**) e un elemento **Row** (**rowElement**), deve aggiungere i valori contenuti in **Row** in una nuova **DataRow**; per fare questo, occorre creare la nuova **DataRow** tramite il metodo **NewRow** della **DataTable**, selezionare tutti gli elementi **ColumnValue** contenuti nell'elemento **Row** e, per ognuno di questi, recuperare la **DataColumn** dalla **DataTable** (se la **DataColumn** non esiste sollevare un'eccezione), trasformare la stringa contenuta nell'elemento **ColumnValue** (proprietà **InnerText**) nel valore di tipo corrispondente al tipo della **DataColumn** (proprietà **DataType**) ed inserirlo nella **DataRow** oppure, se l'attributo di nome **null** vale "true" inserire nella **DataRow** il valore **DBNull.Value**. Al termine dell'operazione inserire la **DataRow** creata nella **DataTable** (metodo **Add** della collezione **Rows**). **Attenzione**: poiché durante il caricamento non è possibile garantire l'integrità referenziale del **DataSet**, è indispensabile disabilitare il controllo dei vincoli prima di effettuare il caricamento e riabilitarlo dopo aver effettuato il caricamento (proprietà **EnforceConstraints** del **DataSet**).

#### DataSetLoader

<pre>+ LoadData ( [in] dataSet : DataSet , [in] fileName : string ) - AddRow ( [in] table : DataTable , [in] rowElement : XmlElement )</pre>
--