

# Default Table Model

José Jarones Bueno

# Clase Alumno

```
public class Alumno {  
  
    private String nombre;  
    private String curso;  
  
    public Alumno(String nombre, String curso) {  
        this.nombre = nombre;  
        this.curso = curso;  
    }  
  
    public String getNombre() {  
        return nombre;  
    }  
  
    public void setNombre(String nombre) {  
        this.nombre = nombre;  
    }  
  
    public String getCurso() {  
        return curso;  
    }  
  
    public void setCurso(String curso) {  
        this.curso = curso;  
    }  
  
}
```

# Clase LogicaNegocio

```
package com.mycompany.alumnostablemodel;

import java.util.ArrayList;
import java.util.List;

/**
 *
 * @author jaron
 */
public class LogicaNegocio {

    public List<Alumno> listaAlumnos = new ArrayList<>();

    public LogicaNegocio() {

        listaAlumnos = new ArrayList<>();
        listaAlumnos.add(new Alumno("Jose", "DAM2"));
        listaAlumnos.add(new Alumno("Eduardo", "DAM2"));
        listaAlumnos.add(new Alumno("Javier", "DAM2"));

    }

    public List<Alumno> getListAlumnos() {
        return listaAlumnos;
    }

}
```

# Creamos AlumnosTableModel

La clase hereda de la clase abstracta AbstractTableModel y ponemos que se creen todos los métodos abstractos que debemos implementar.

```
    */  
    package com.mycompany.alumnosTableModel;  
  
    import javax.swing.table.AbstractTableModel;  
  
    /**  
     *  
     * @author jaron  
     */  
    public class AlumnosTableModel extends AbstractTableModel {  
  
        @Override  
        public int getRowCount() {  
              
        }  
  
        @Override  
        public int getColumnCount() {  
              
        }  
  
        @Override  
        public Object getValueAt(int rowIndex, int columnIndex) {  
              
        }  
  
    }
```

## Creo el constructor (paso el listado de alumnos)

```
public class AlumnosTableModel extends AbstractTableModel {  
    private List<Alumno> listAlumno;  
  
    public AlumnosTableModel(List<Alumno> listAlumno){  
  
        this.listAlumno = listAlumno;  
    }  
}
```

# Implementamos los métodos

```
@Override
public int getRowCount() {
    return listAlumno.size();
}
```

```
@Override
public int getColumnCount() {
    return 2;
}
```

```
@Override
public Object getValueAt(int rowIndex, int columnIndex) {

    switch (columnIndex){
        case 0:
            return listAlumno.get(rowIndex).getNombre();
        case 1:
            return listAlumno.get(rowIndex).getCurso();
    }

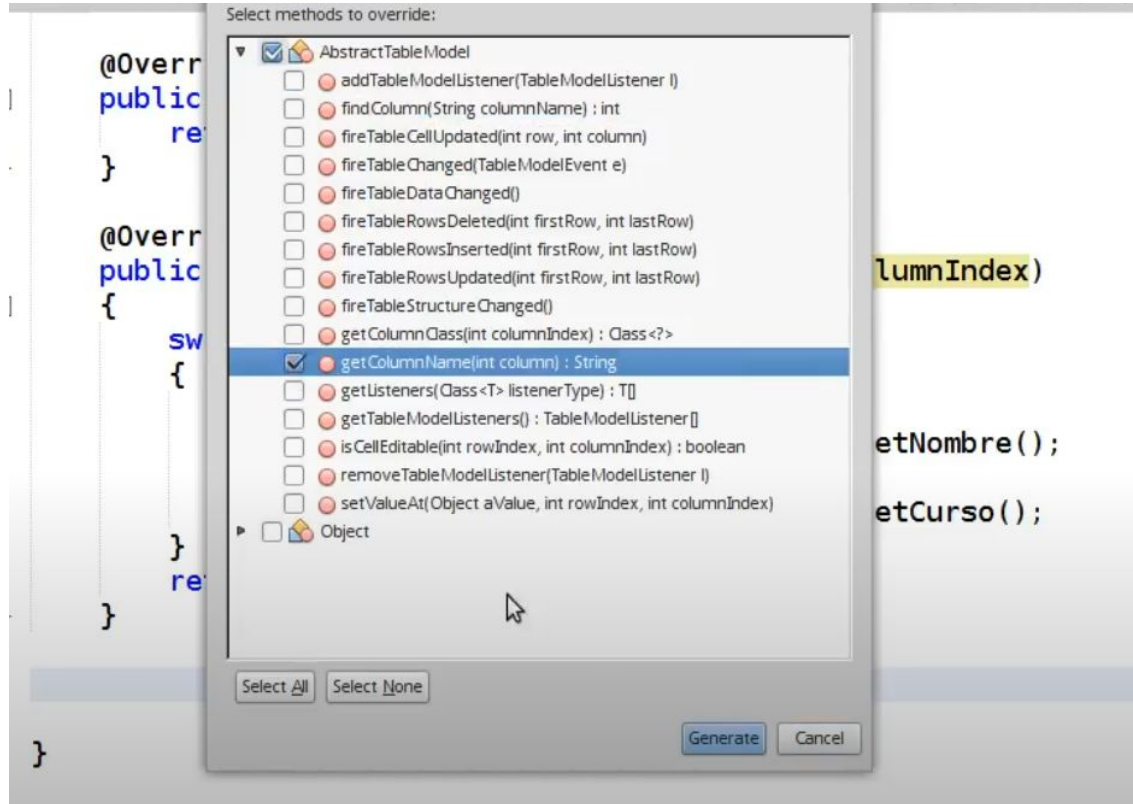
    return null;
}
```

# En VentanaPrincipal añadimos el modelo

```
1 2 /
3 public class PantallaPrincipal extends javax.swing.JFrame {
4
5     private LogicaNegocio logicaNegocio=new LogicaNegocio();
6
7     /**
8      * Creates new form PantallaPrincipal
9      */
10    public PantallaPrincipal() {
11        initComponents();
12        this.jTable1.setModel(new AlumnosTableModel(logicaNegocio.getListaAlumnos()));
13    }
14 }
```

# Para el nombre de las columnas

Vamos a sobrecargar un método: getColumnNave





# Creamos un array con los nombres de las columnas

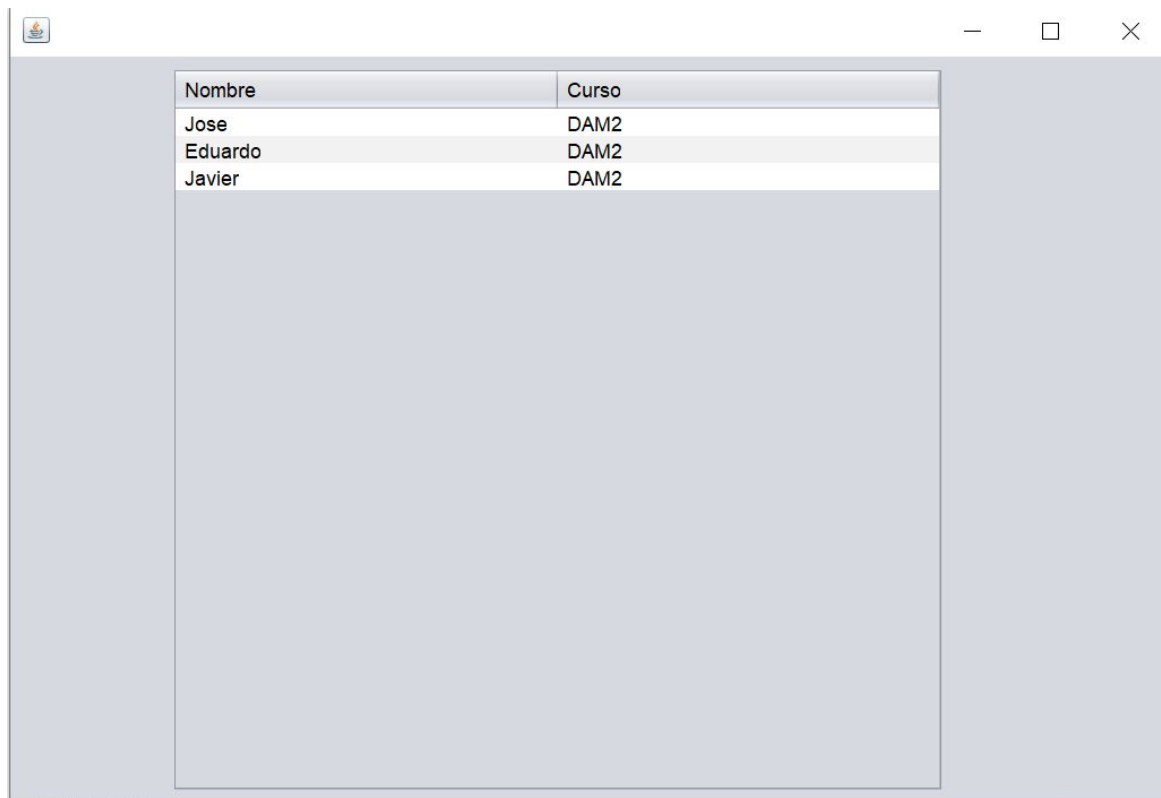
```
private String[] columnas = {"Nombre", "Curso"};
```



# Y lo devolvemos en el método

```
@Override  
public String getColumnName(int column) {  
    return columnas[column];  
}
```

# Resultado



A screenshot of a software application window. The window has a title bar with a small icon on the left and standard minimize, maximize, and close buttons on the right. The main content area displays a table with two columns: 'Nombre' and 'Curso'. The table contains three rows of data. The background of the window is a light gray, and the table has a white header and alternating light gray and white rows for data.

Nombre	Curso
Jose	DAM2
Eduardo	DAM2
Javier	DAM2

## Tarea 6 - Tabla Coches

Crea un nuevo programa que cumpla con las siguientes características:

- Una clase Coches, con los atributos: Modelo, Color, Precio, FechaMatriculacion
- Una ventana principal donde se muestren los coches en un JTable
- Un DefaultTableModel propio creado para la tabla coches
- Una LogicaNegocio donde se añada un listado de 6 coches.

