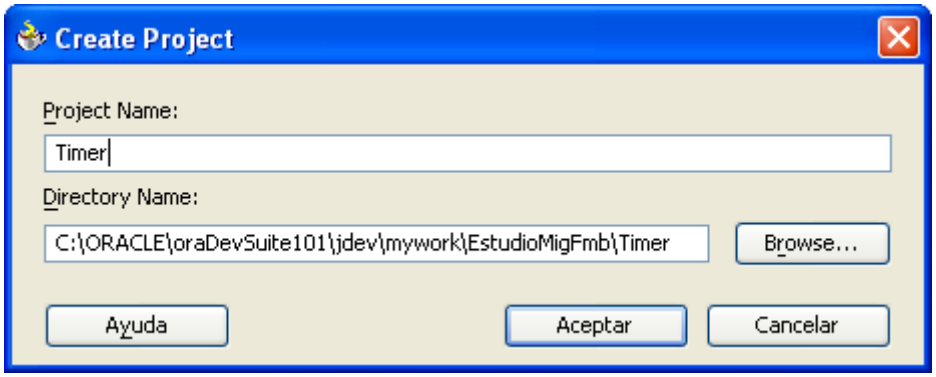
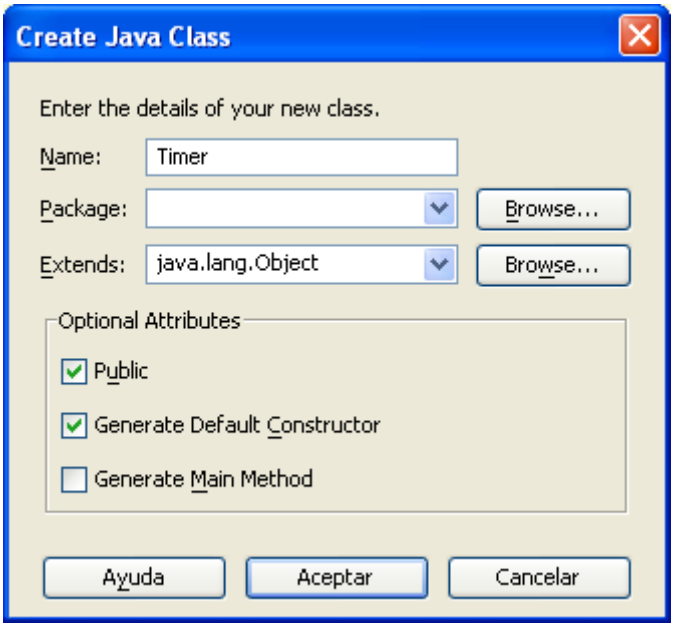
Mi primer JavaBean paso a paso

1. Crear una nueva aplicación desde JDeveloper

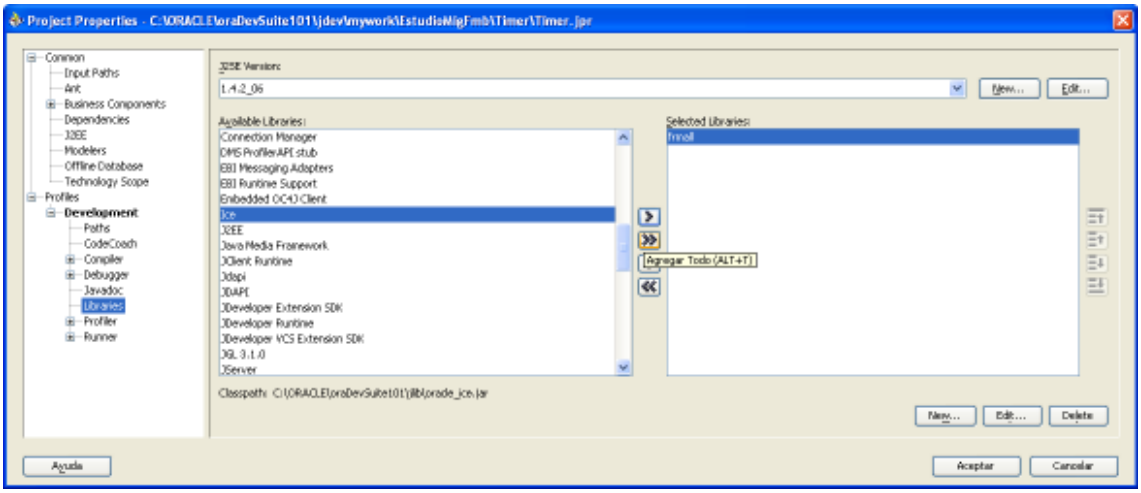
2. Crear un nuevo proyecto



3. Crear una nueva clase Java



4. Incorporar a nuestro proyecto la librería frmall.jar



5. Plantilla para implementar nuestra clase Java

```
import oracle.forms.ui.VBean;

import oracle.forms.properties.ID;

import oracle.forms.handler.IHandler;

import oracle.forms.ui.CustomEvent;

import java.lang.Runnable;


public class Timer extends VBean implements Runnable

{

  static IHandler mHandler;

…….

  // setters y getters

  protected static final ID POWER = ID.registerProperty("POWER");

  protected static final ID TIME = ID.registerProperty("TIME");

  protected static final ID REPEAT = ID.registerProperty("REPEAT");


  //Valor enviado al formulario cuendo timer expira

  protected static final ID AVISOTIMEREXPIRADO   =
ID.registerProperty("AVISOTIMEREXPIRADO");

……

  //Constructor por defecto

  public Timer()

  {

    super();

  }


  //Inicialización

  public void init (IHandler handler)

  {

    super.init(handler);

    mHandler = handler;

  }
```

```
      //Setters
      public boolean setProperty(ID property, Object value)
      {
        if(property == POWER)
        {
          //Obtenemos el valor de la propiedad
          String sParam = (String)value ;
    ...……..

      //Getters
      public Object getProperty(ID property)
      {
        if (property == TIME)
        {
    ……….
      //Envío de un mensaje al formulario
      public void dispatch_event()
      {
        CustomEvent ce = new CustomEvent(mHandler, AVISOTIMEREXPIRADO);
        dispatchCustomEvent(ce);
    }
    …..
```

a. Declaración de las propiedades de nuestra clase

```
protected static final ID POWER = ID.registerProperty("POWER");
protected static final ID TIME = ID.registerProperty("TIME");
protected static final ID REPEAT = ID.registerProperty("REPEAT");
```

b. Método que inicializa la clase

```
Init
```

c. Se dispara cuando desde el formulario utilizamos la p.u. Set_Custom_Property

```
setProperty
```

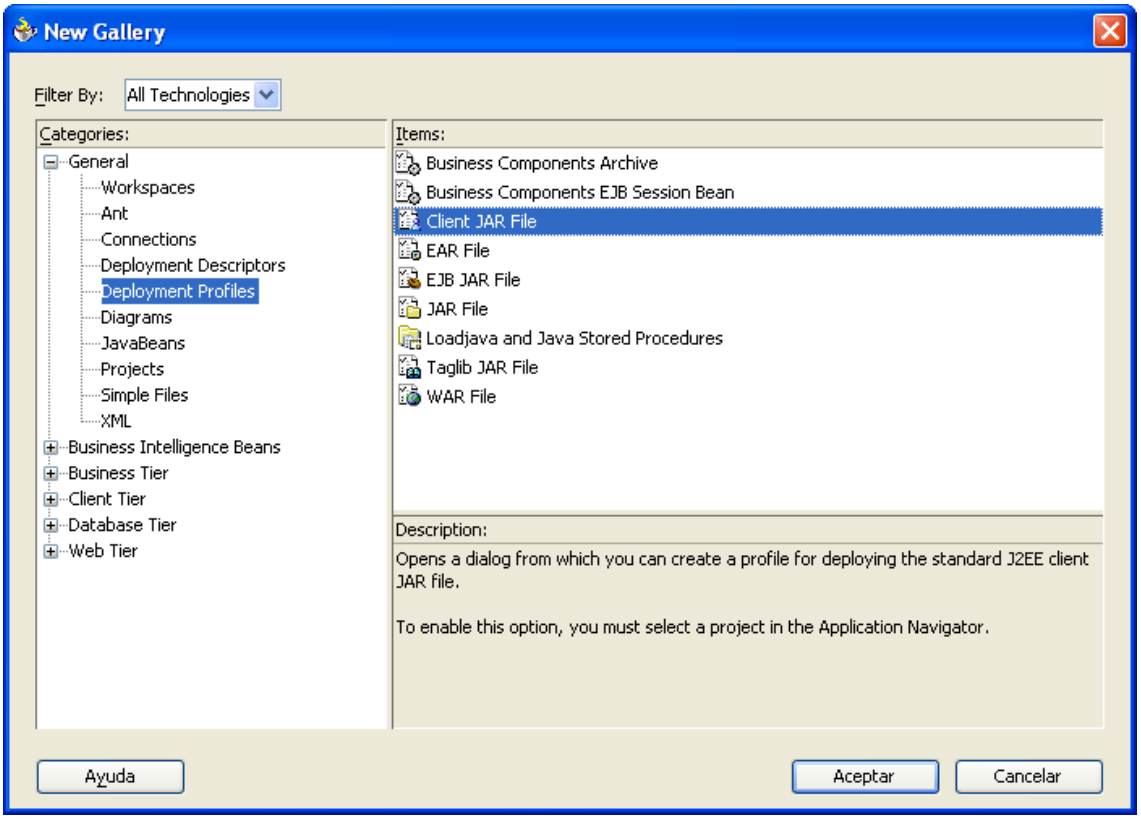
d. Se dispara cuando desde el formulario utilizamos la p.u. get_Custom_Property

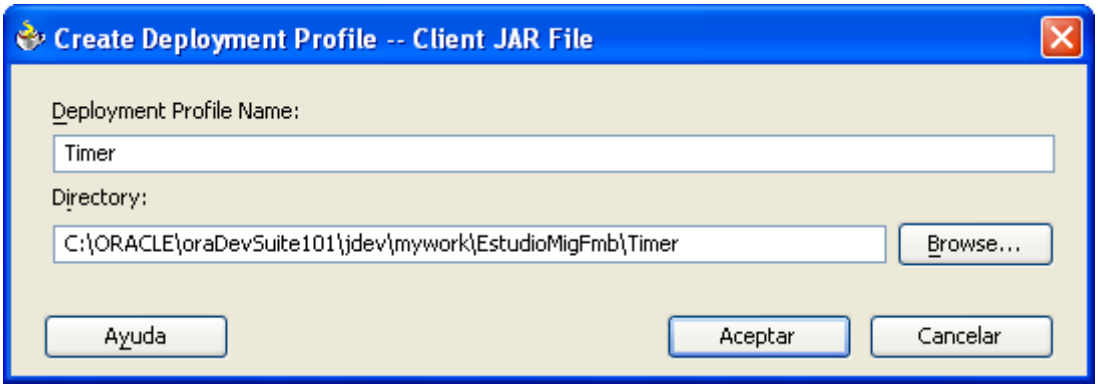
getProperty

e. Evento que se utiliza dede la clase JAVA para cominicarse con el formulario. En el formulario se dispara el trigger WHEN-CUSTOM-ITEM-EVENT del item tipo “Bean Area” donde se ha asignado la clase Timer.jar.

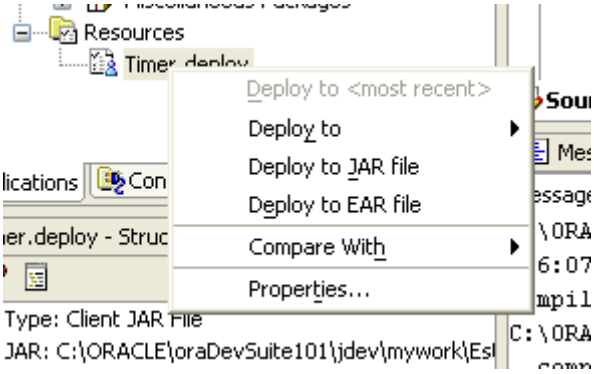
dispatch_event

6. Una vez implementada nuestra clase y testeada creamos un .jar





7. Deploy to JAR file



8. Como utilizar el JavaBean desde nuestro formulario

a. (set) Para asignar un valor a una propiedad de nuestro JavaBean

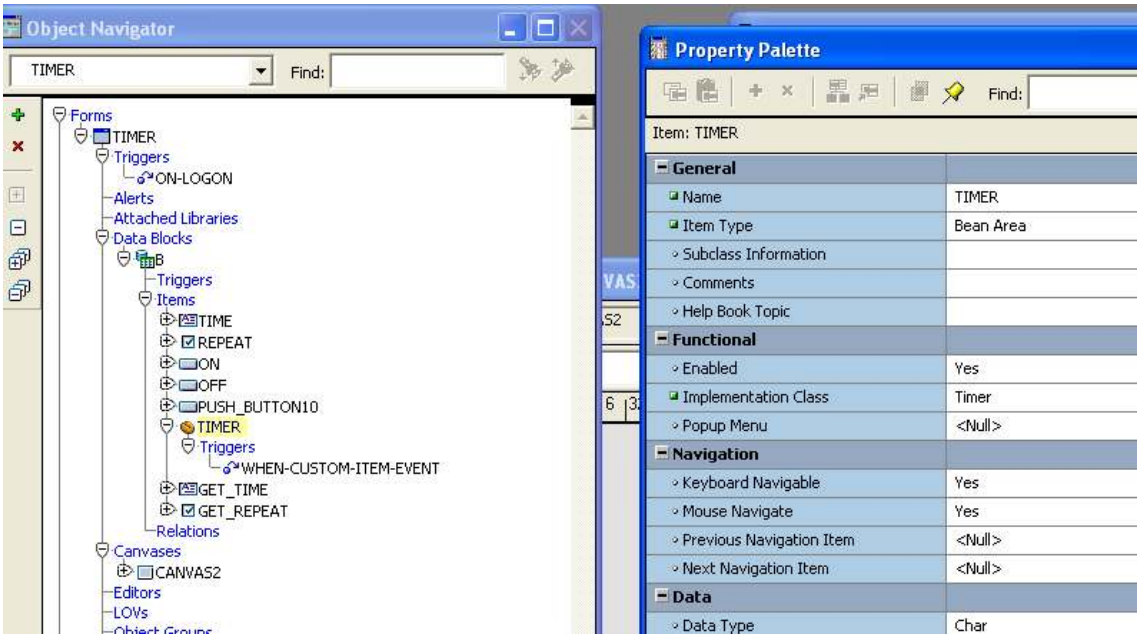
Set_Custom_Property('item_name', record_number, 'property_name', 'property_value') ;

b. (get) Para recuperar el valor de una propiedad de nuestro JavaBean

Varchar2 := Get_Custom_Property('item_name', record_number, 'property_name') ;

c. Crear un item tipo "Bean Area" y asignar el nombre de nuestra clase

A tener en cuenta. El item tipo BeanArea ha de tener CANVAS asignado



d. Crear trigger WHEN-CUSTOM-ITEM-EVENT en el item tipo “Bean Area”. Este se dispara en nuestro ejemplo cada vez que el timer expira.

e. Crear los ítems para que el usuario pueda decidir el tiempo en ms y si el timer se ha de repetir o no.

f. Crea un botón para arrancar el Timer

```
set_custom_property('B.TIMER', 1, 'TIME', :TIME);

set_custom_property('B.TIMER', 1, 'REPEAT', :REPEAT);

set_custom_property('B.TIMER', 1, 'POWER', 'START');
```

g. Crear un botón para parar el timer

```
set_custom_property('B.TIMER', 1, 'POWER', 'STOP');
```

h. Crear un botón para recuperar valores de propiedades de nuestro timer

```
 :GET_TIME := get_custom_property('B.TIMER', 1, 'TIME');

 :GET_REPEAT := get_custom_property('B.TIMER', 1, 'REPEAT');
```

9. Copiar fichero Timer.jar en <ORACLE_HOME>\forms\Java

10. Formsweb.cfg

```
archive_jini=…,Timer.jar
```

11. El resultado final