

POO con Cocoa

Primera Aplicación: *HolaMundo*

Amilcar Meneses Viveros
ameneses@computacion.cs.cinvestav.mx

Sección de Computación
Departamento de Ingeniería Eléctrica
CINVESTAV-IPN

Objetivos

- Verificar el modelo Sender-Target
- Arquitectura de la aplicación
- Jerarquía *View*
- Diseño de la aplicación
- Introducción a las aplicaciones
ProjectBuilder e InterfaceBuilder

Modelo Sender-Target

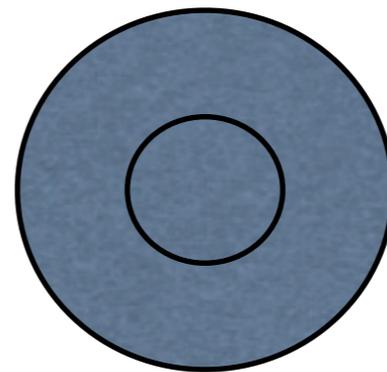
- Permite desencadenar acciones (enviar mensajes) desde objetos gráficos
- Interfaz: - `(IBAction)doAction:(id)sender;`

SENDER

Botón 1

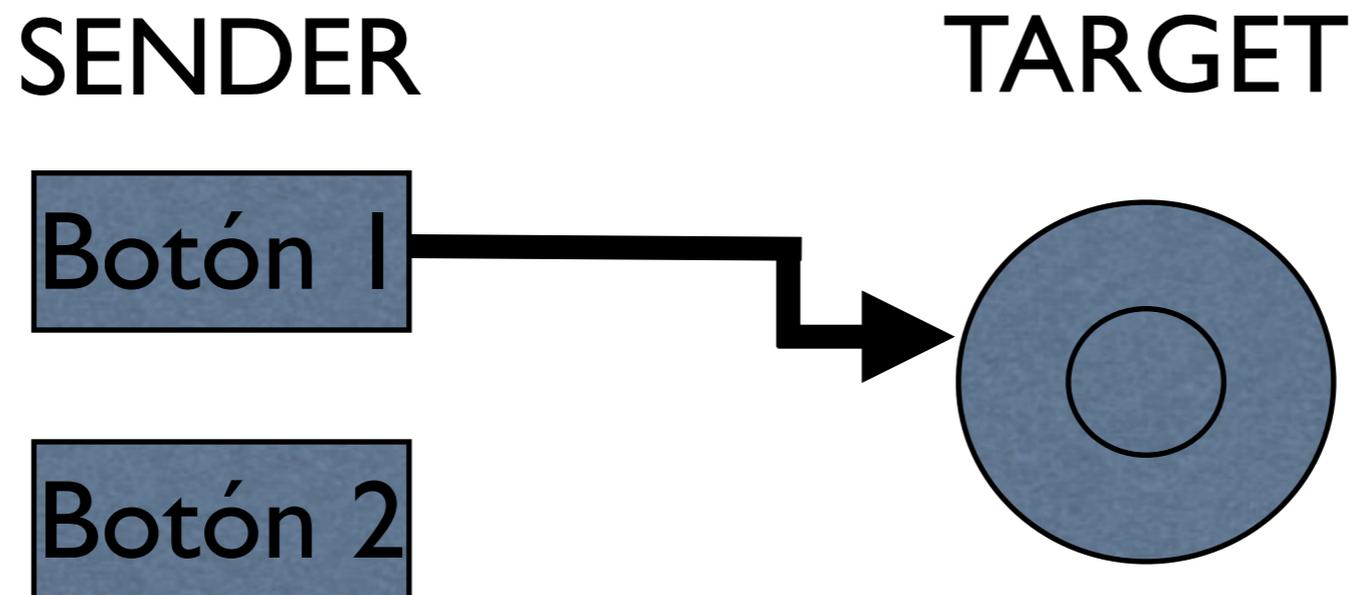
Botón 2

TARGET



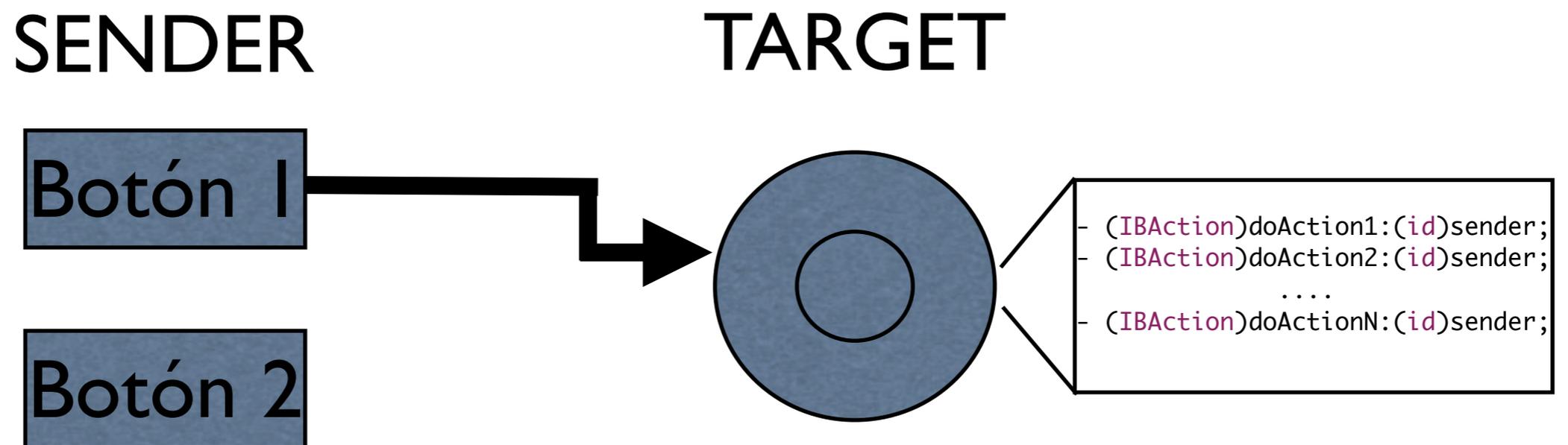
Modelo Sender-Target

- Permite desencadenar acciones (enviar mensajes) desde objetos gráficos
- Interfaz: - `(IBAction)doAction:(id)sender;`



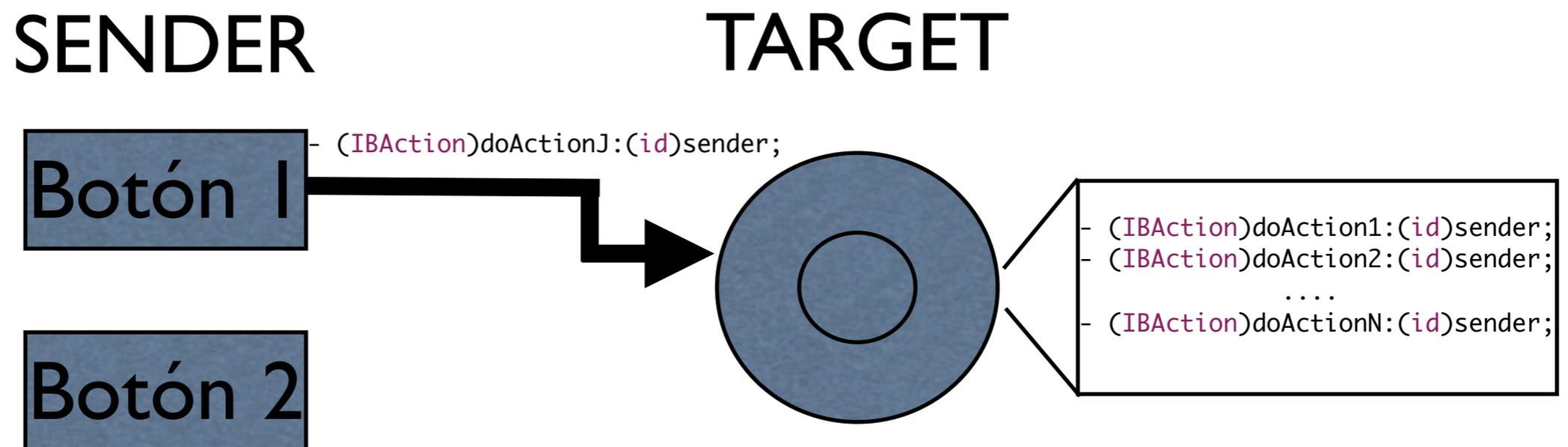
Modelo Sender-Target

- Permite desencadenar acciones (enviar mensajes) desde objetos gráficos
- Interfaz: - `(IBAction)doAction:(id)sender;`



Modelo Sender-Target

- Permite desencadenar acciones (enviar mensajes) desde objetos gráficos
- Interfaz: - `(IBAction)doAction:(id)sender;`

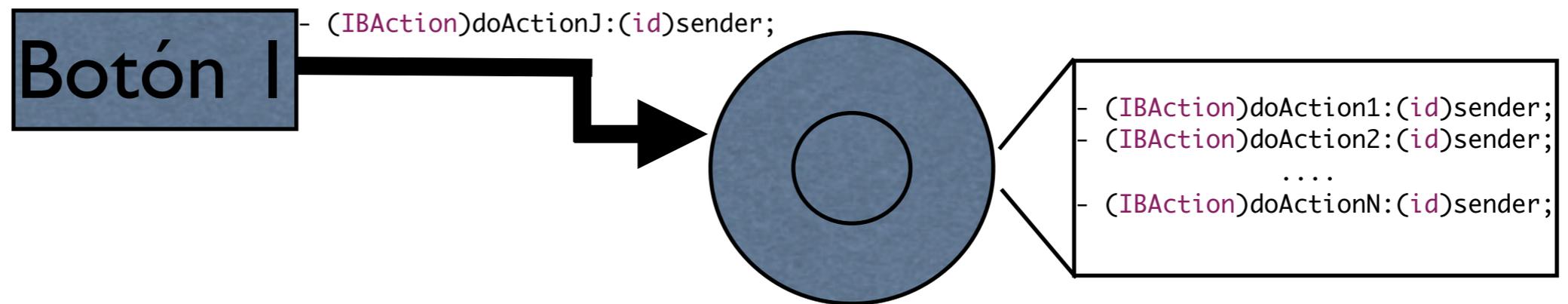


Modelo Sender-Target

- Permite desencadenar acciones (enviar mensajes) desde objetos gráficos
- Interfaz: - (IBAction)doAction:(id)sender;

SENDER

TARGET



```
[target performSelector: @selector(doActionJ:)  
withObject: self];
```

Modelo Sender-Target

- Cada objeto sender tiene un sólo objeto target
- Un objeto target puede recibir mensajes de varios objetos sender
- Un objeto target puede recibir el mismo mensaje de varios objetos sender

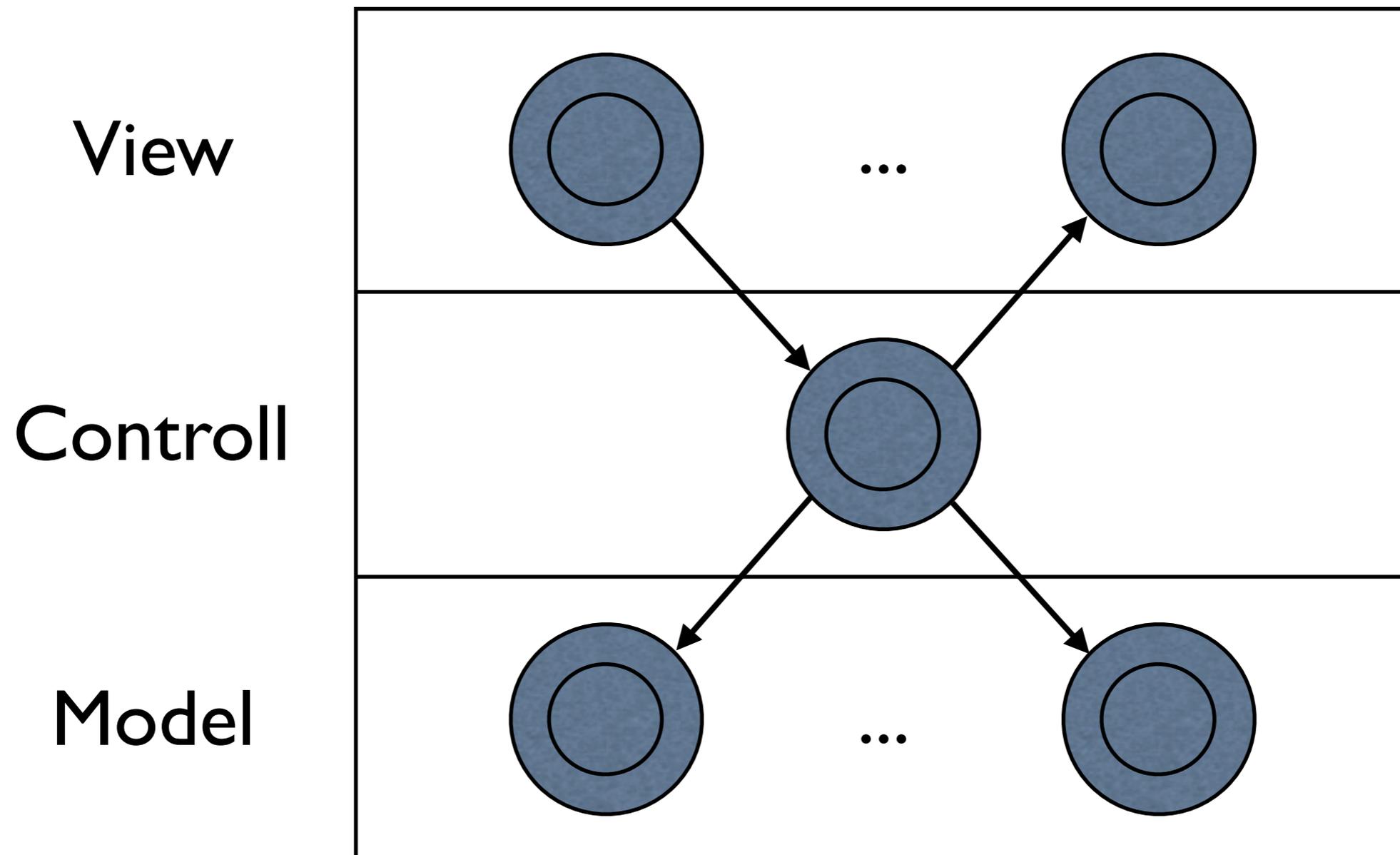
Diseño

Model-Controll-View

- **VIEW:** Interfaz gráfica del usuario
 1. Se utiliza para desencadenar eventos.
 2. Presenta la información del usuario. Sólo es responsable de almacenar la información que despliega.
- **CONTROLL:** Liga los objetos View con los Model
- **MODEL:** Representan datos y comportamientos básicos de la aplicación (no interactúan directamente con el usuario).

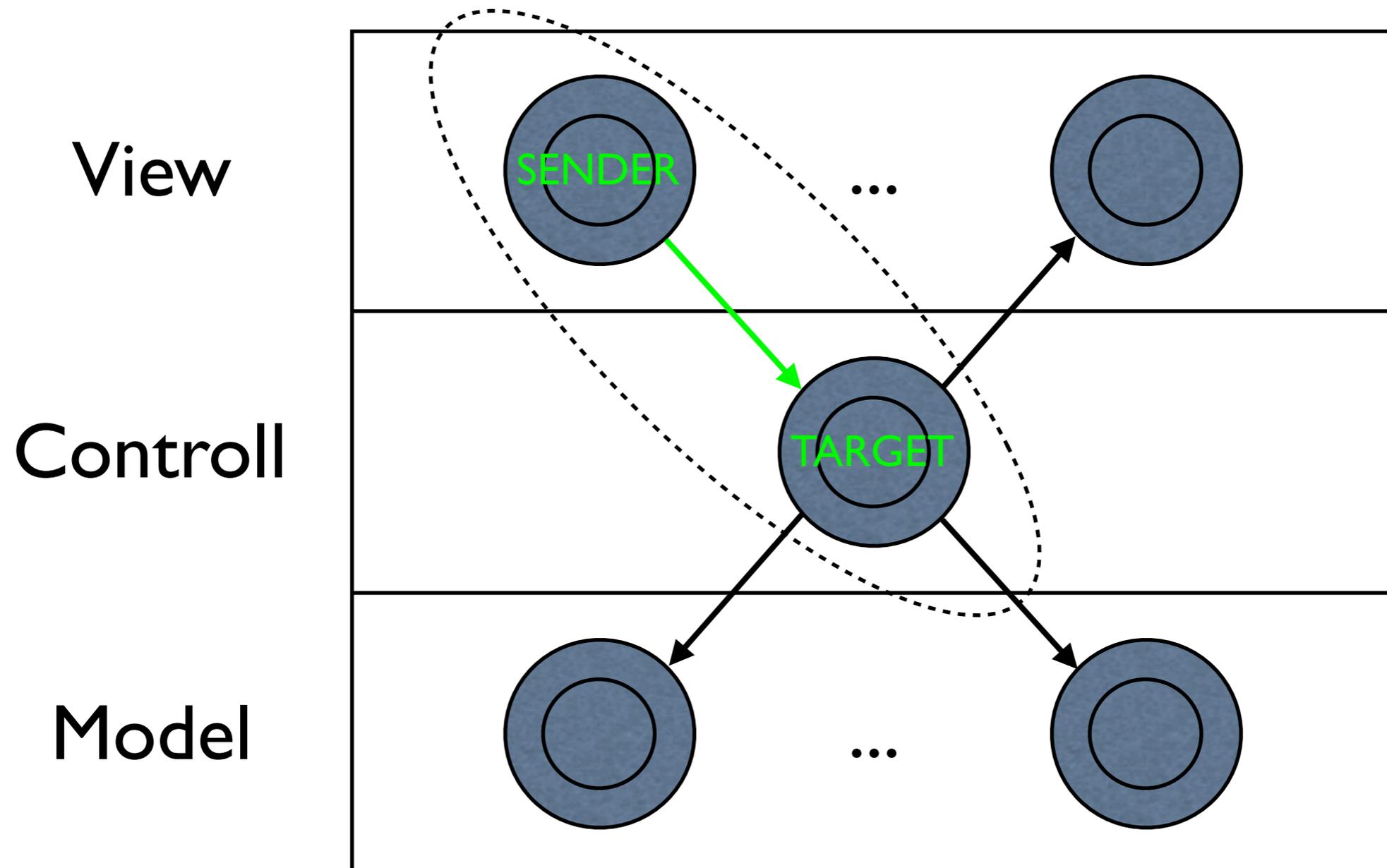
Diseño

Model-Controll-View



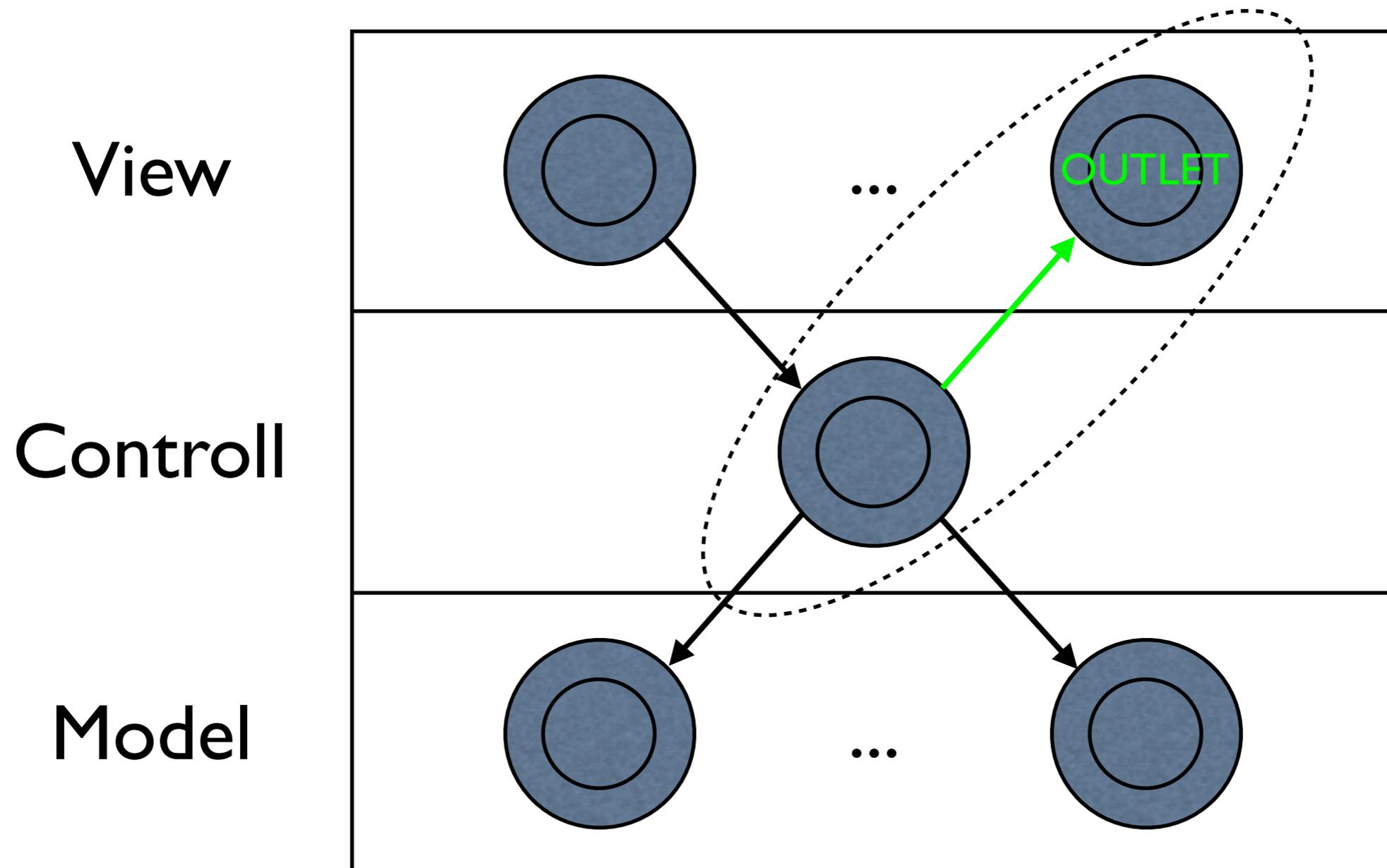
Diseño

Model-Controll-View



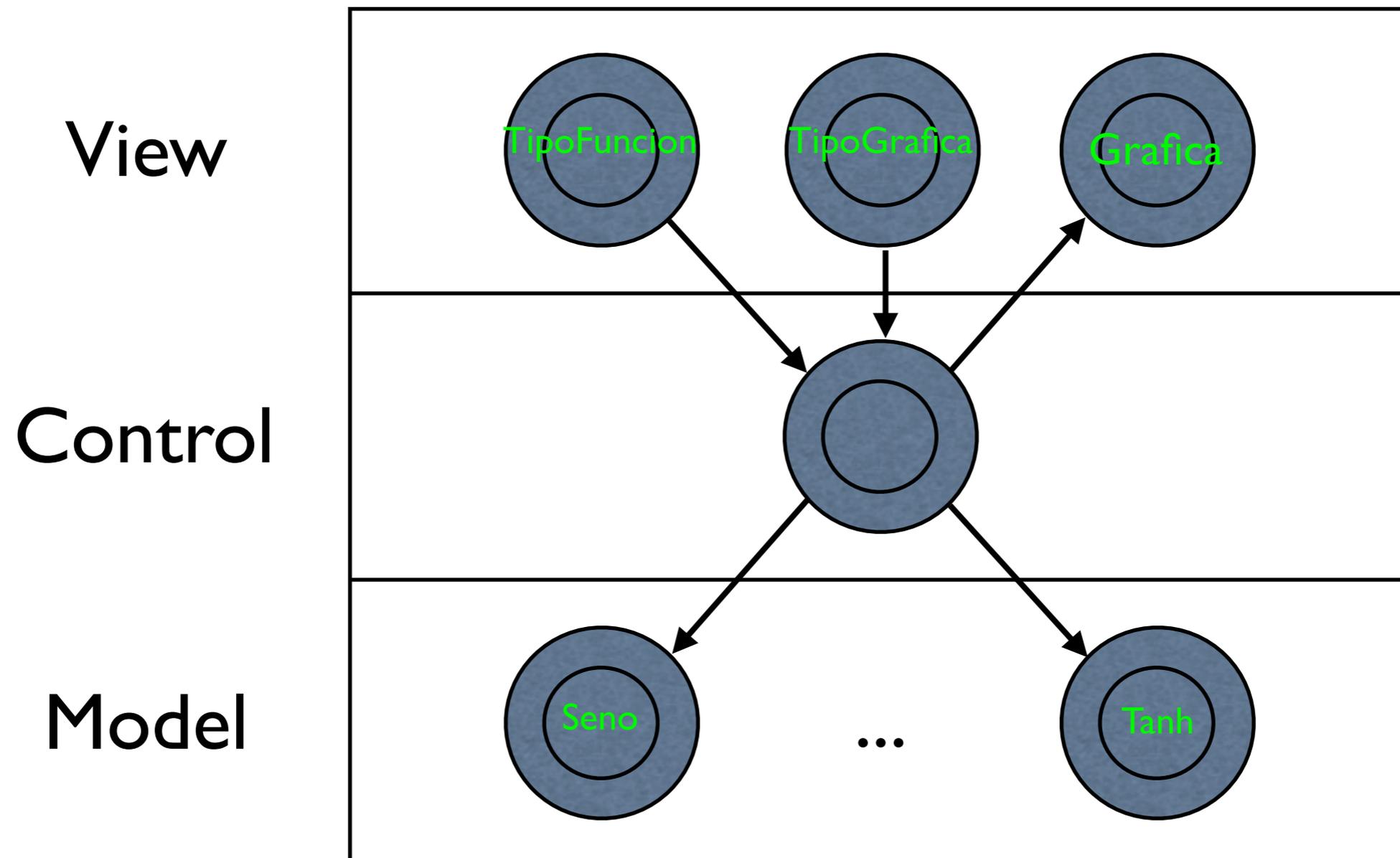
Diseño

Model-Controll-View



Diseño

Model-Control-View



Diseño

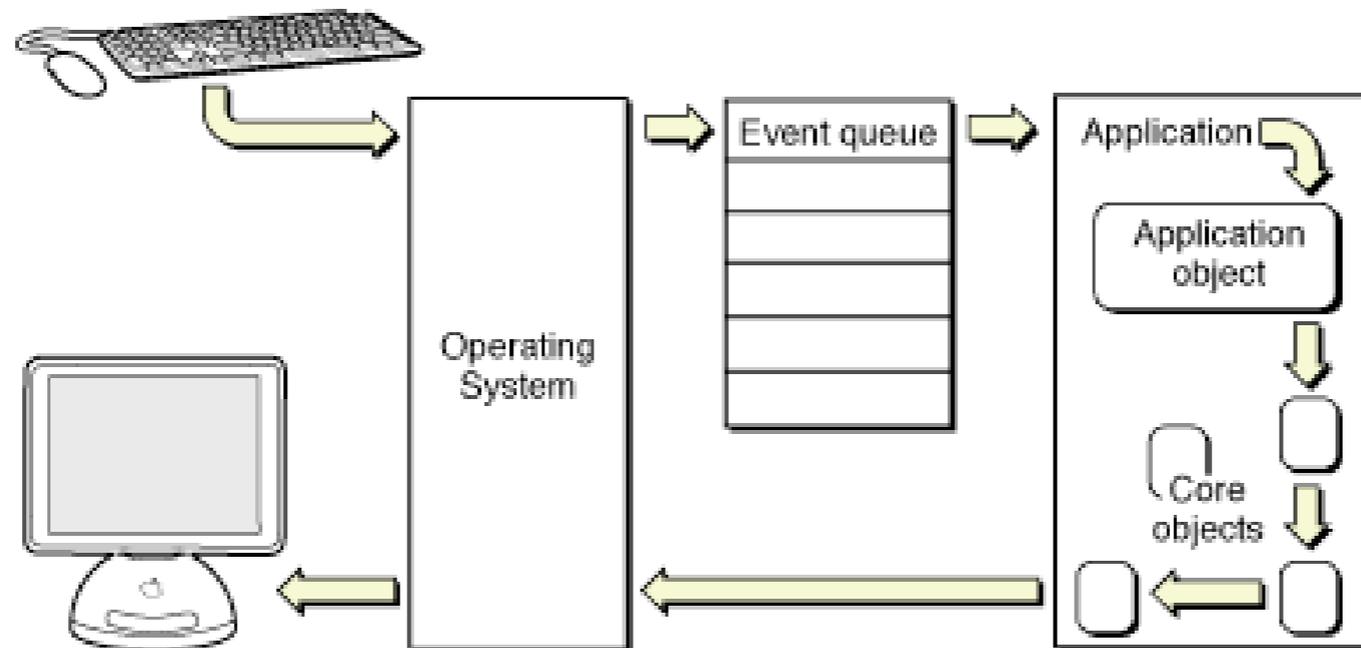
Model-Control-View



Aplicaciones Cocoa

- Uso de XCode para la integración de aplicaciones con mac OS X (*Workspace* y *WindowManager*), via *ProjectBuilder*
- Respuesta a acciones del usuario: abrir, cerrar y miniaturizar ventanas, opción de salir de la aplicación
- Manejo de eventos a través de objetos gráficos de la biblioteca *ApplicationKit* de *Cocoa* y objetos creados por el usuario
- Paradigma para el manejo de eventos: ciclo del evento y jerarquía *View*

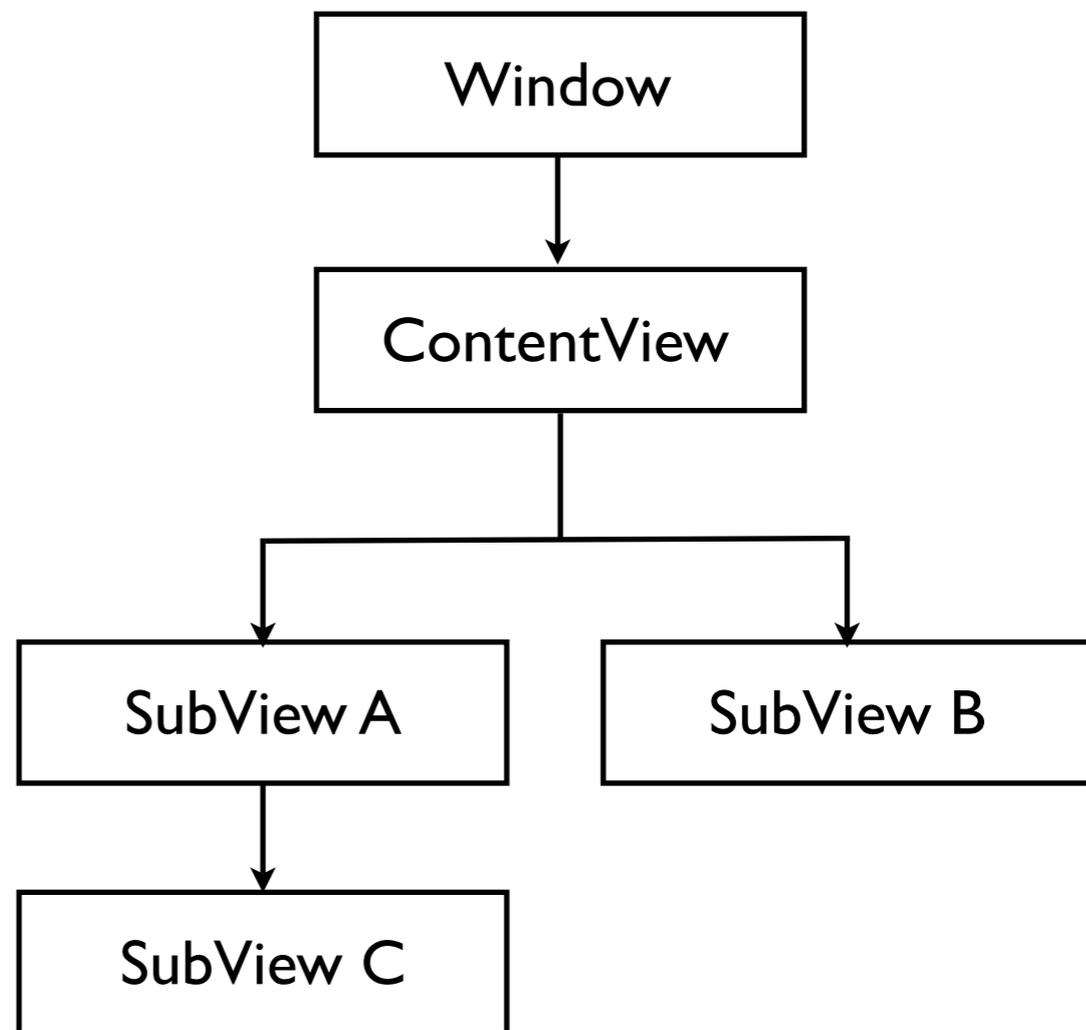
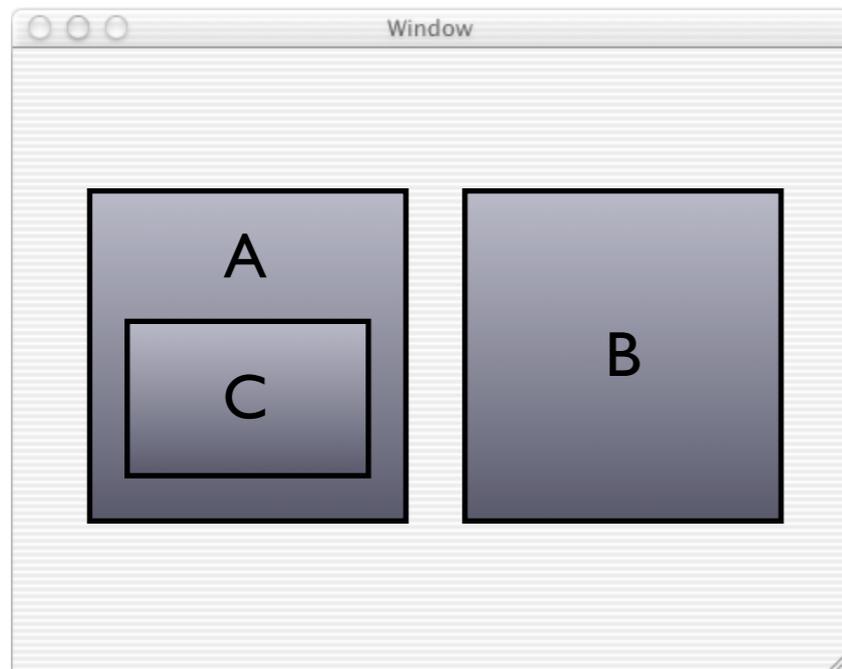
Ciclo del Evento



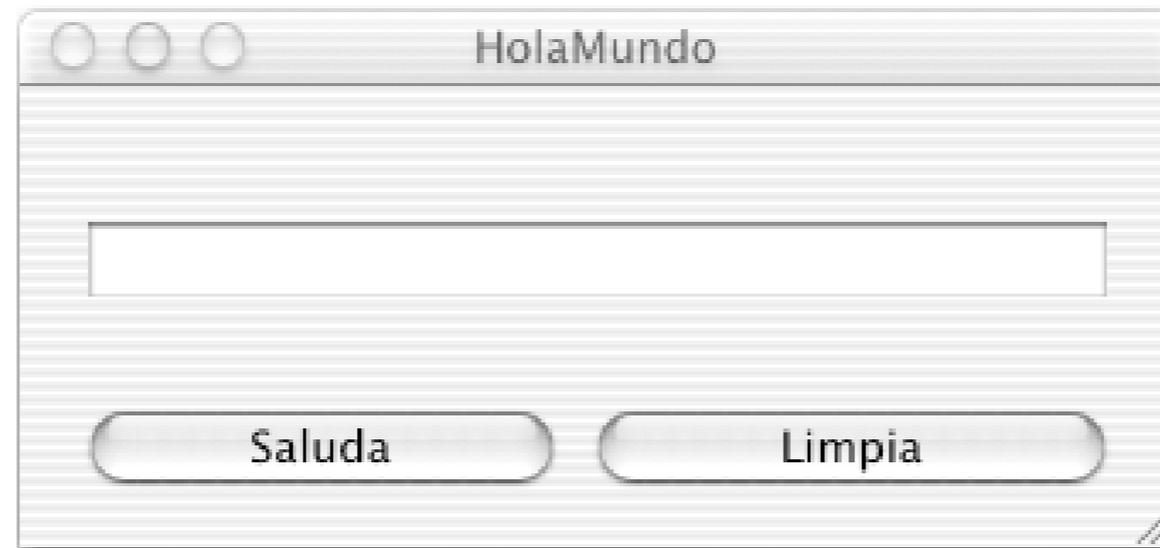
```
#import <AppKit/AppKit.h>
```

```
int main(int argc, const char *argv[]) {  
    return NSApplicationMain(argc, argv);  
}
```

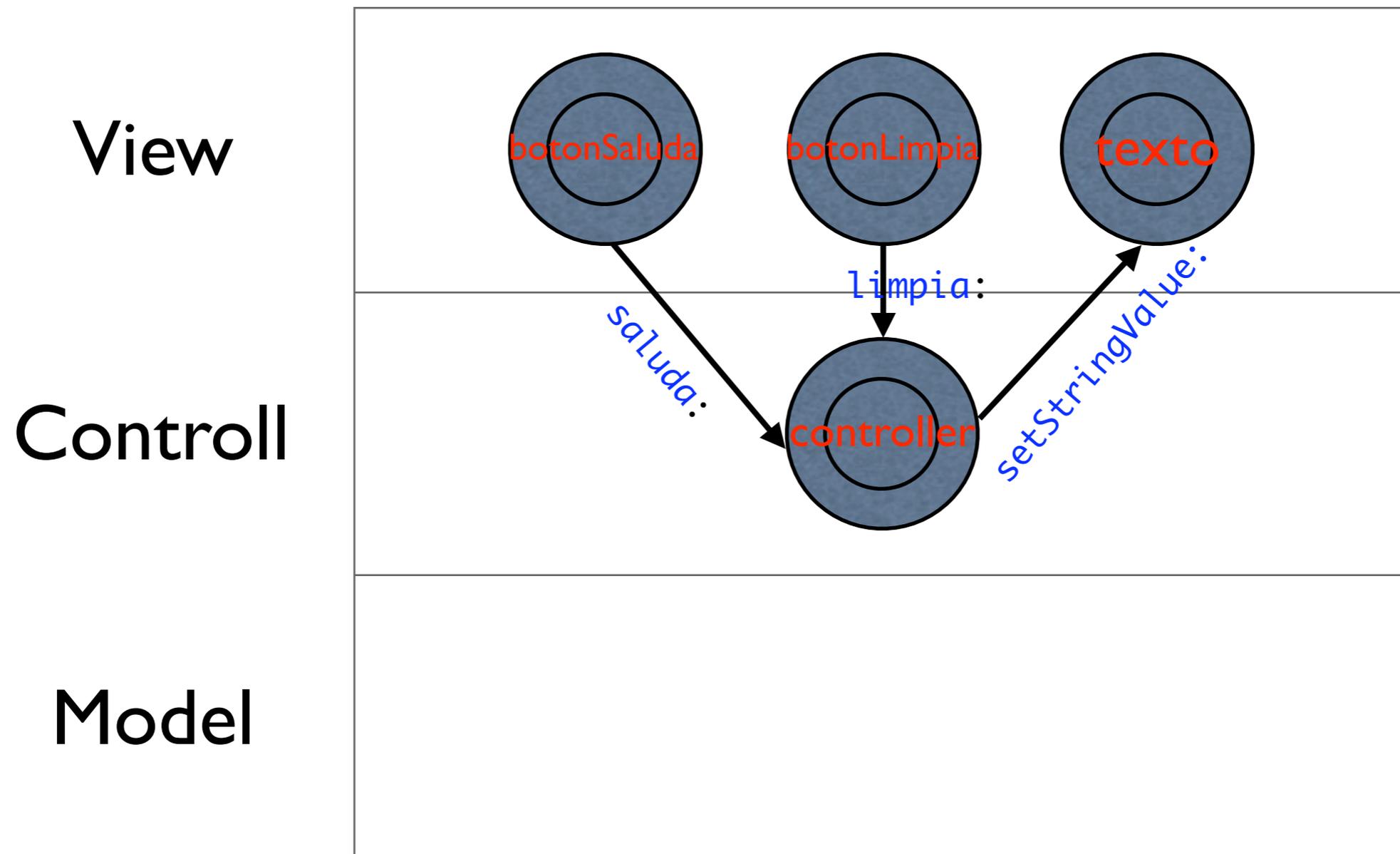
Jerarquía View



Aplicación *HolaMundo*

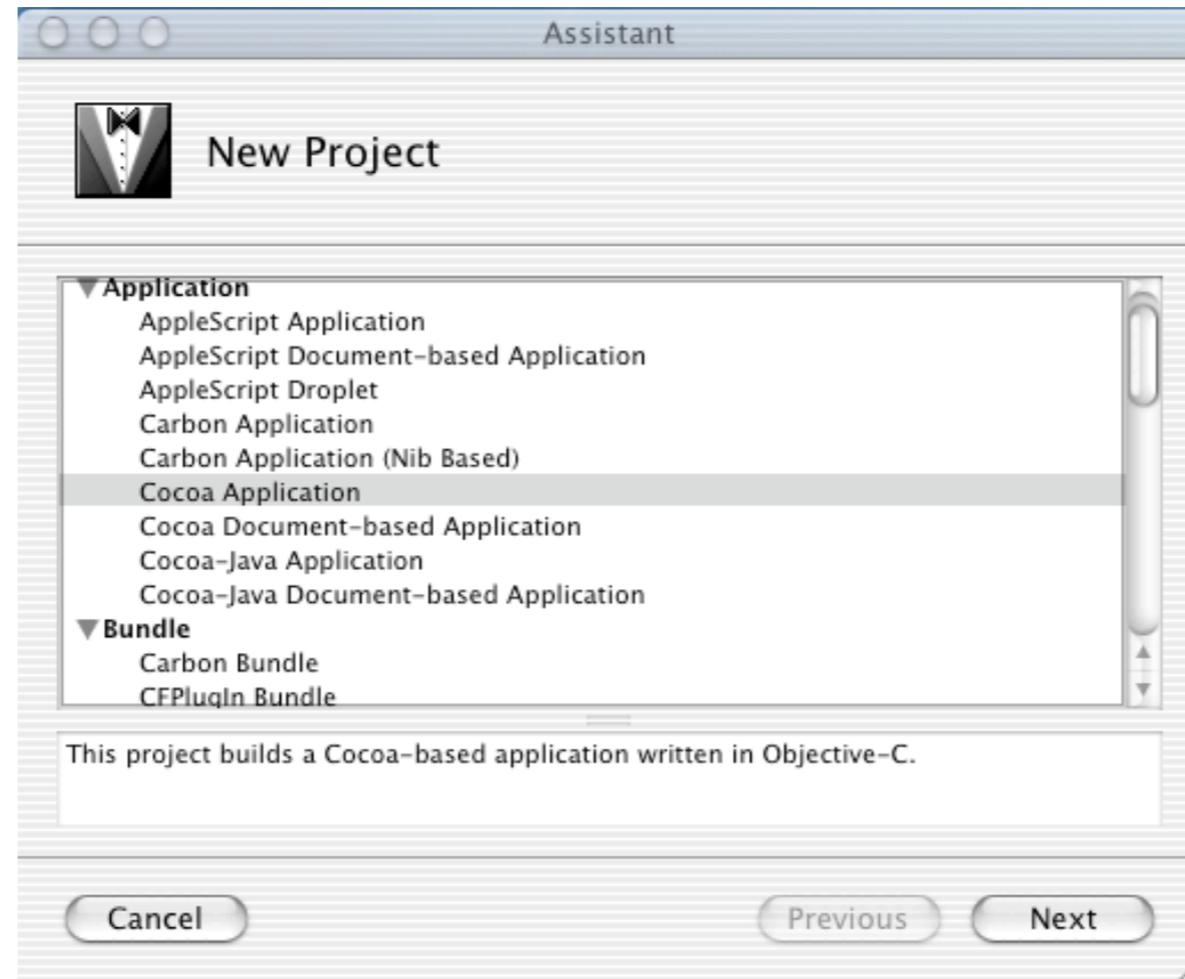


Aplicación *HolaMundo*



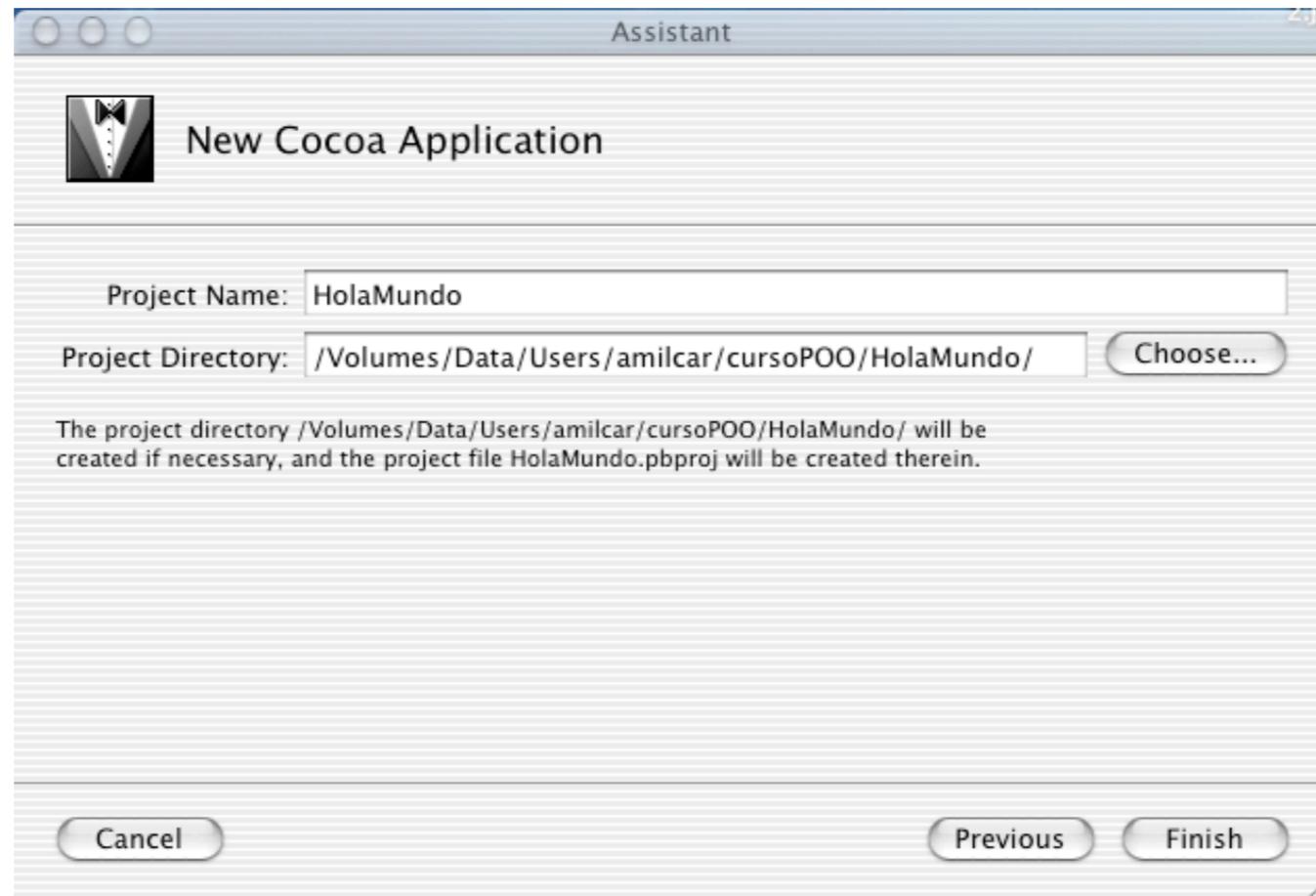
Aplicación *HolaMundo*

- Crear un nuevo proyecto con el *ProjectBuilder*.
Opción New Project del submenú File



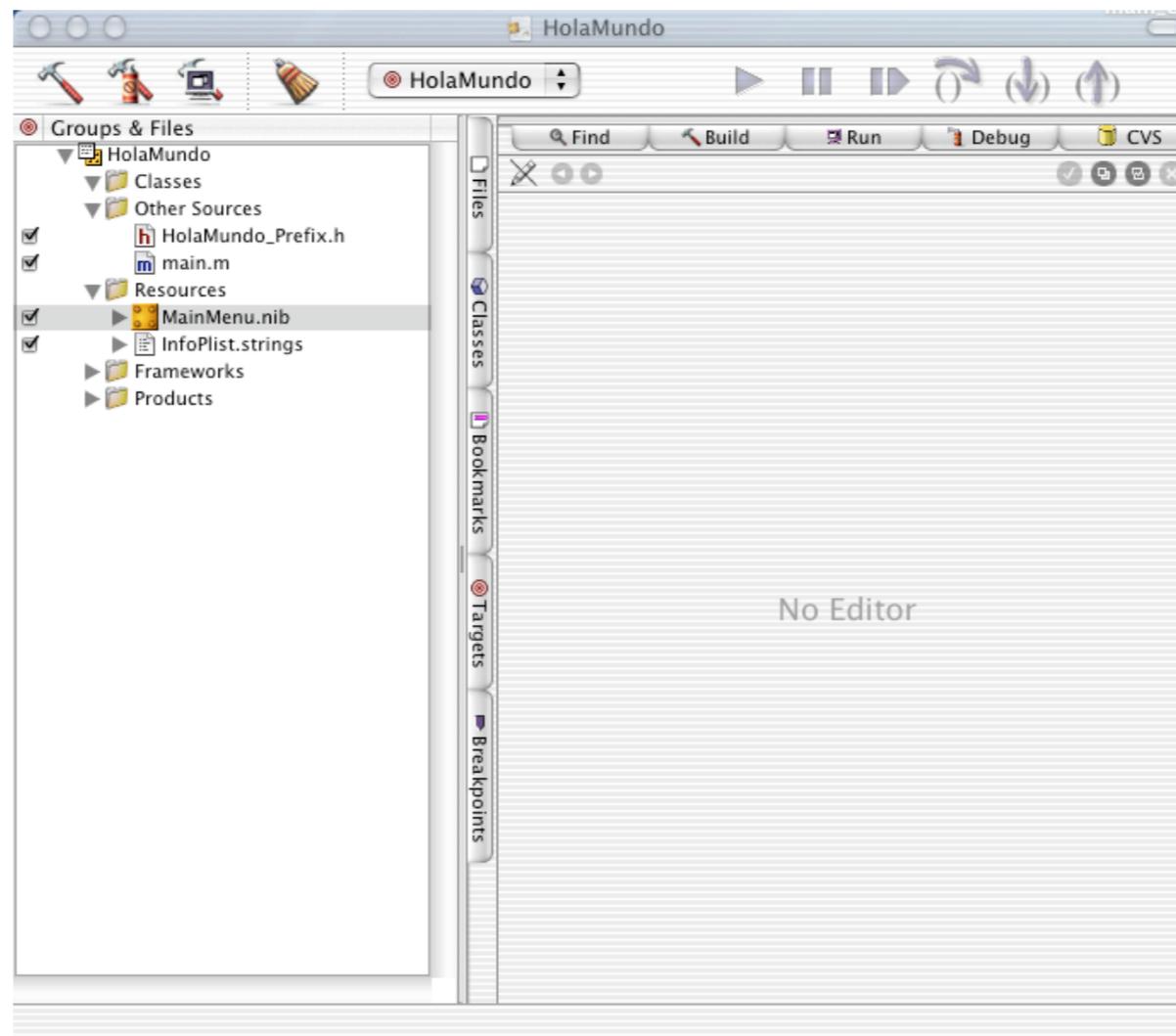
Aplicación *HolaMundo*

- Después seleccionar el nombre de la aplicación (*HolaMundo*) y el lugar donde quedará

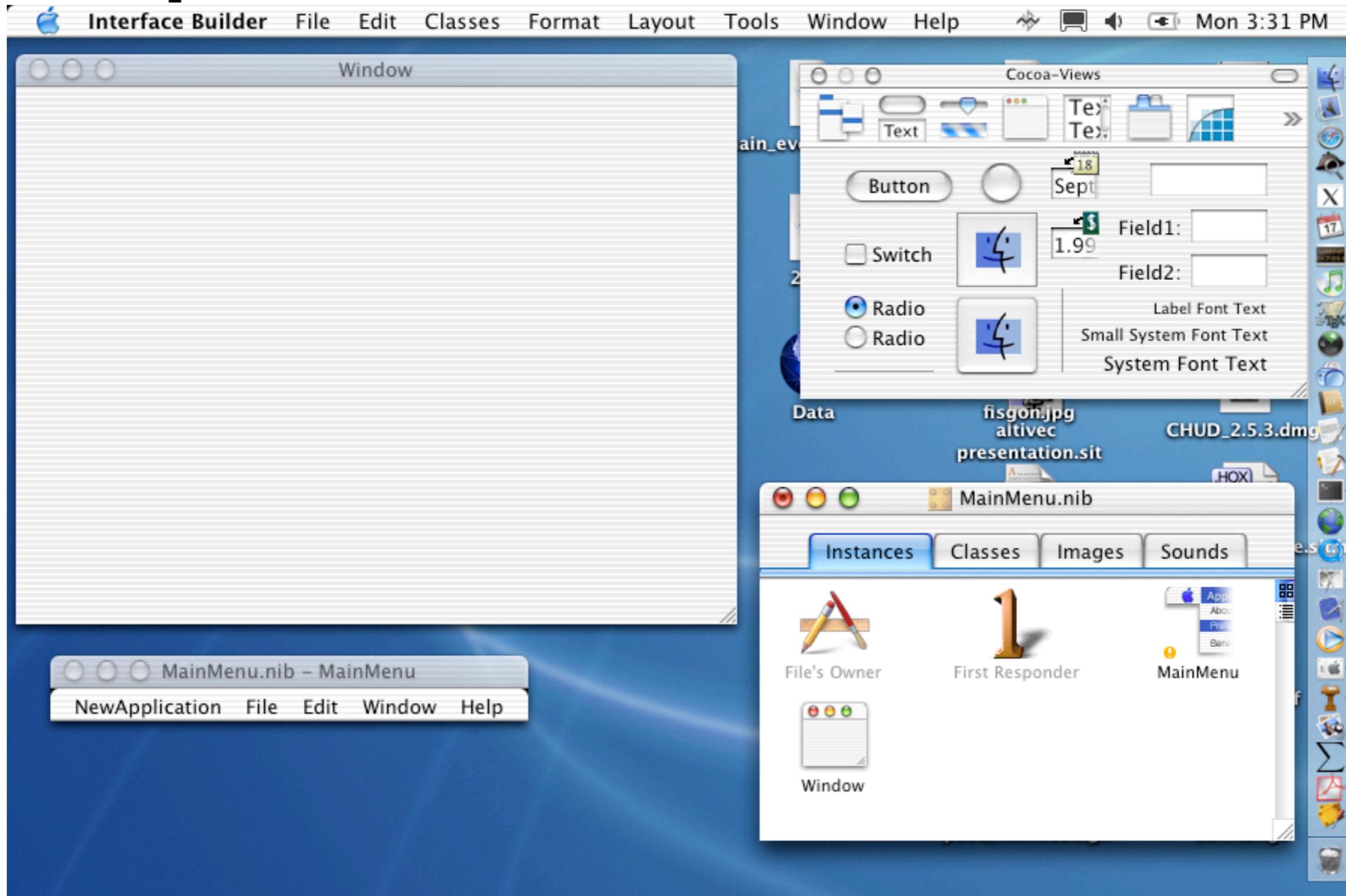


Aplicación *HolaMundo*

- Abrir el archivo asociado a la interfaz gráfica. Esto cargará la aplicación *InterfaceBuilder*

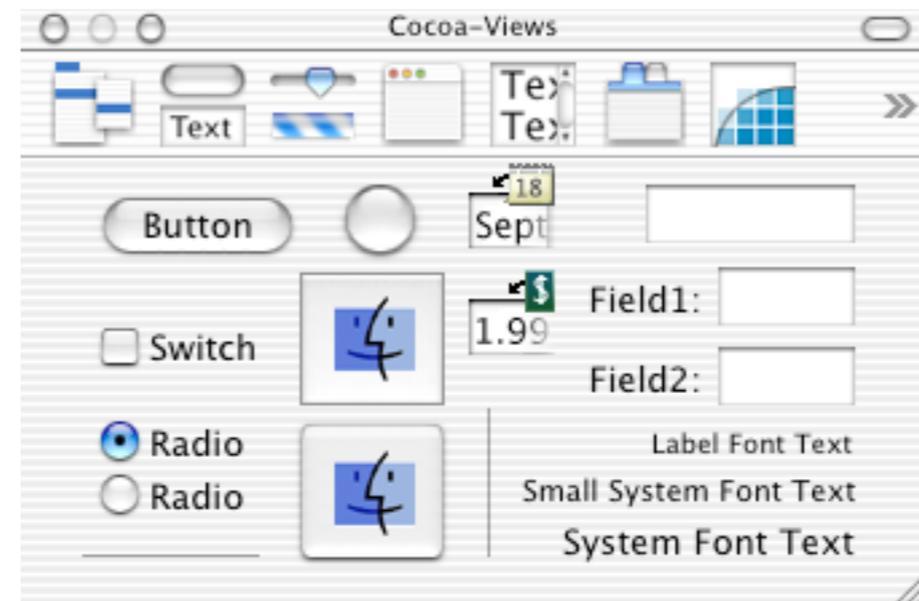


Aplicación *HolaMundo*



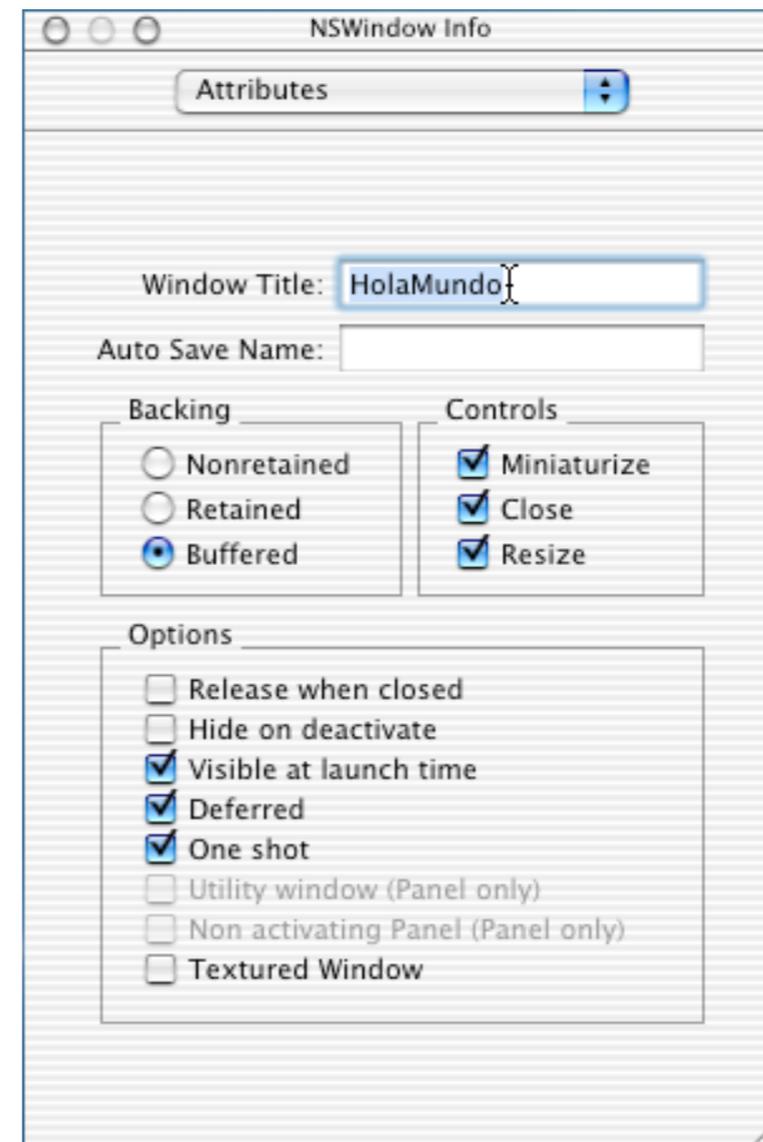
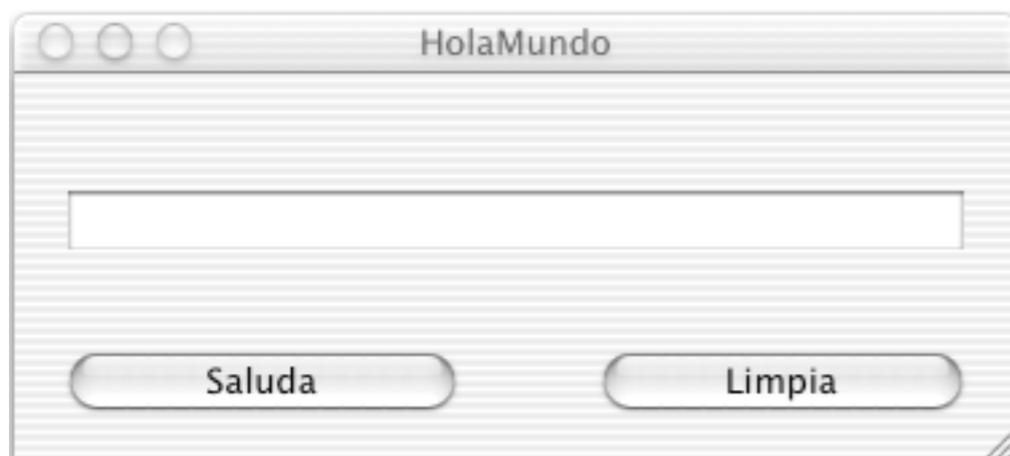
Aplicación *HolaMundo*

- Construir la interfaz gráfica (Drag and Drop) con ayuda de los objetos del panel Palette



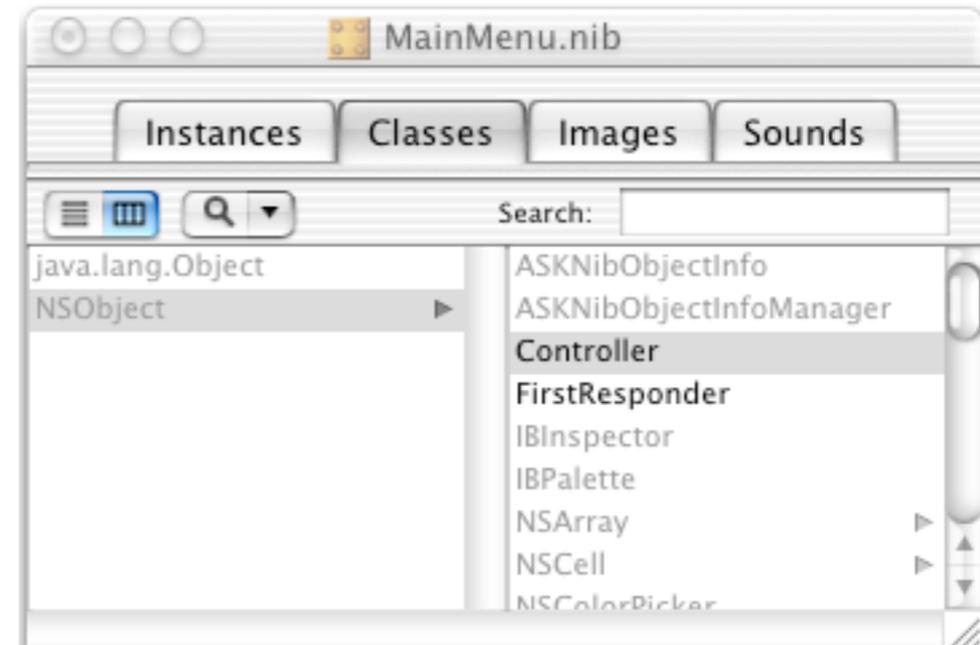
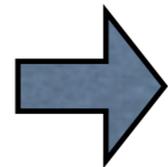
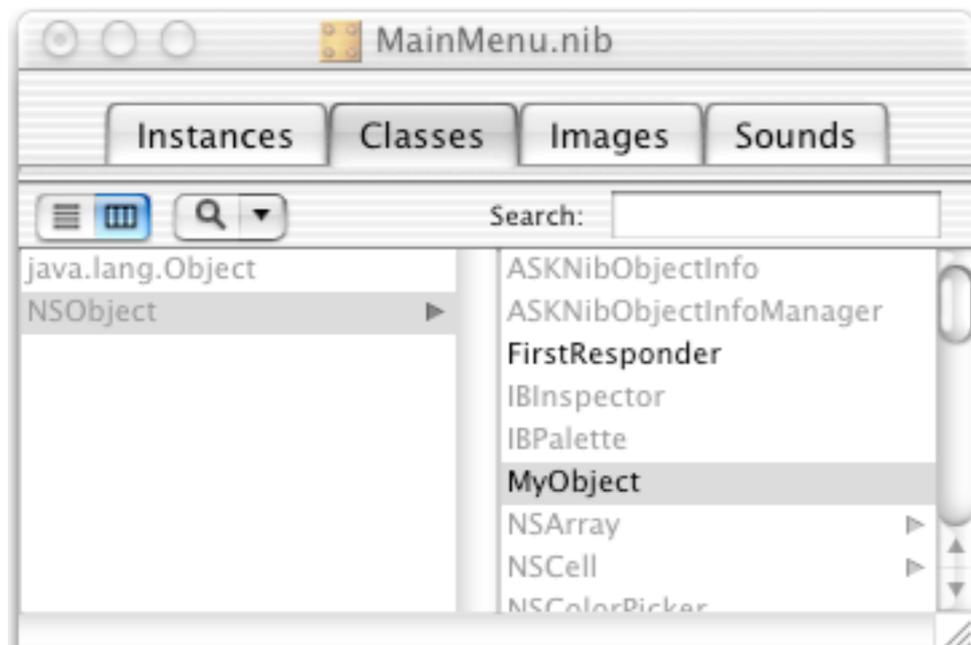
Aplicación *HolaMundo*

- Cambiar en título de la ventana a HolaMundo. Seleccionar la ventana y desde el panel *Info* modificar en el campo *Window Title*



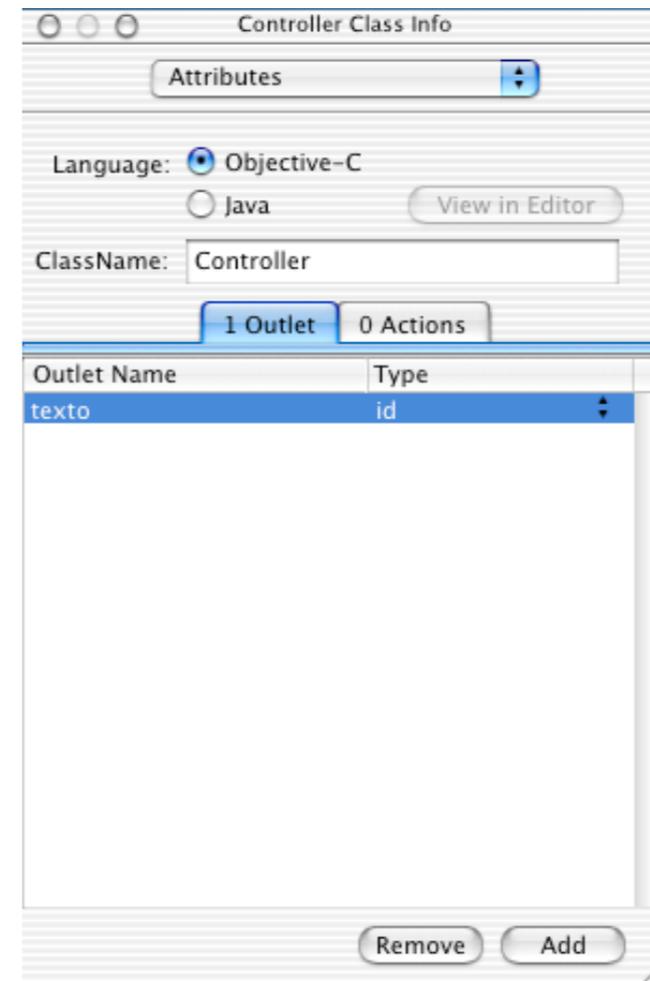
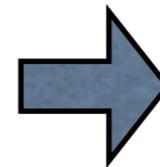
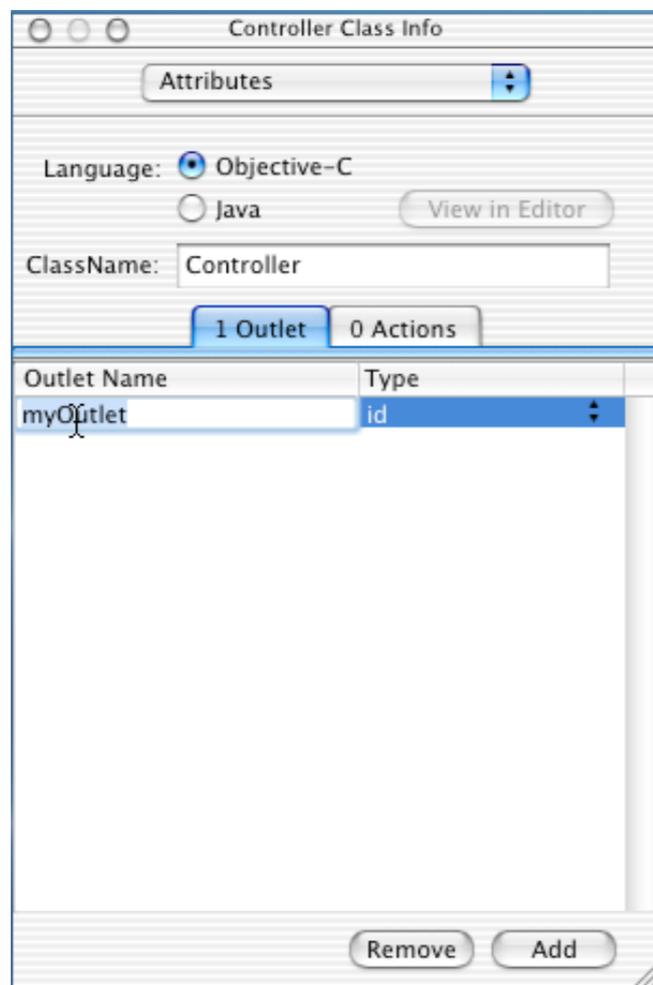
Aplicación *HolaMundo*

- Crear la clase Controller, como subclase de NSObject. Desde la opción Classes de la ventana *MainMenu.nib*



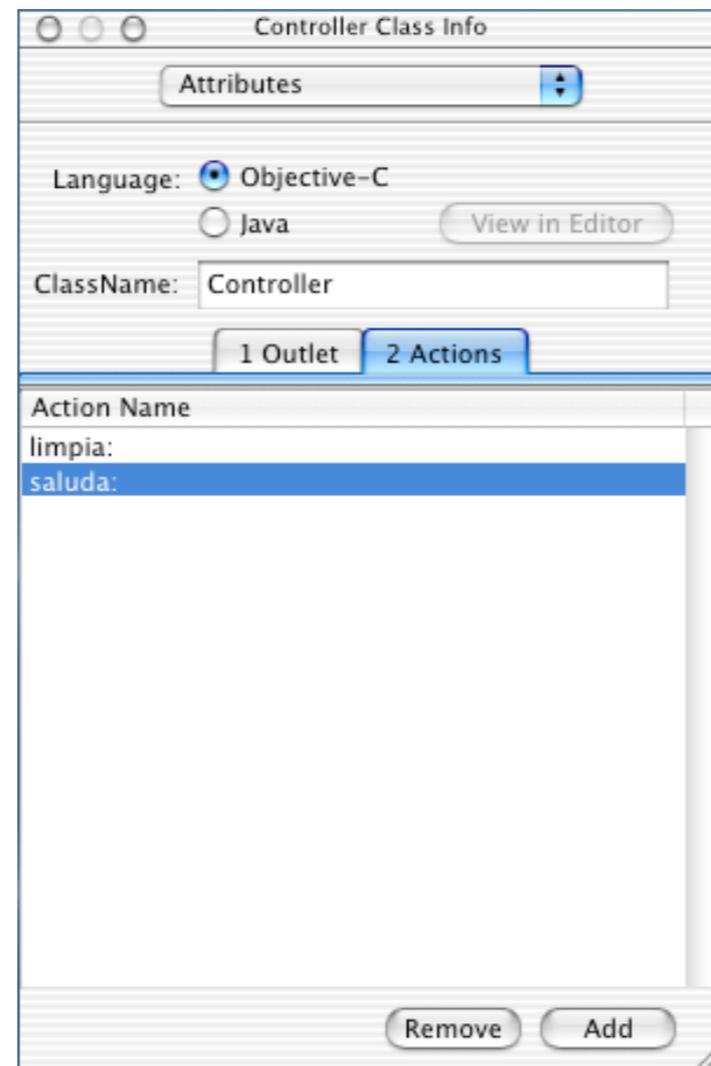
Aplicación *HolaMundo*

- Agregar el *OUTLET* texto. Seleccionar la clase Controller y desde el panel *Info*



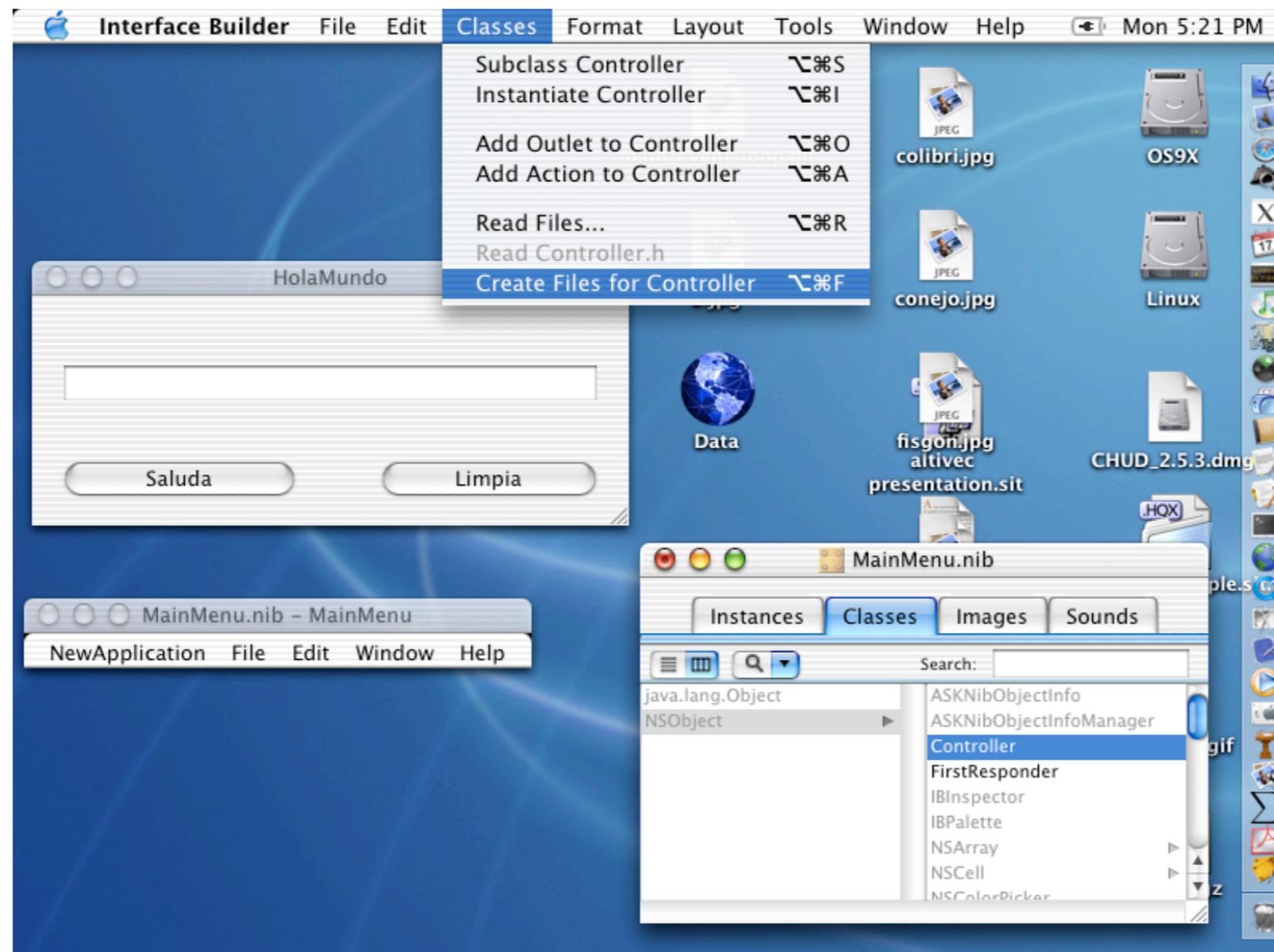
Aplicación *HolaMundo*

- Agregar las **ACTIONS** saluda: y limpia. Seleccionar la clase Controller y desde el panel *Info*



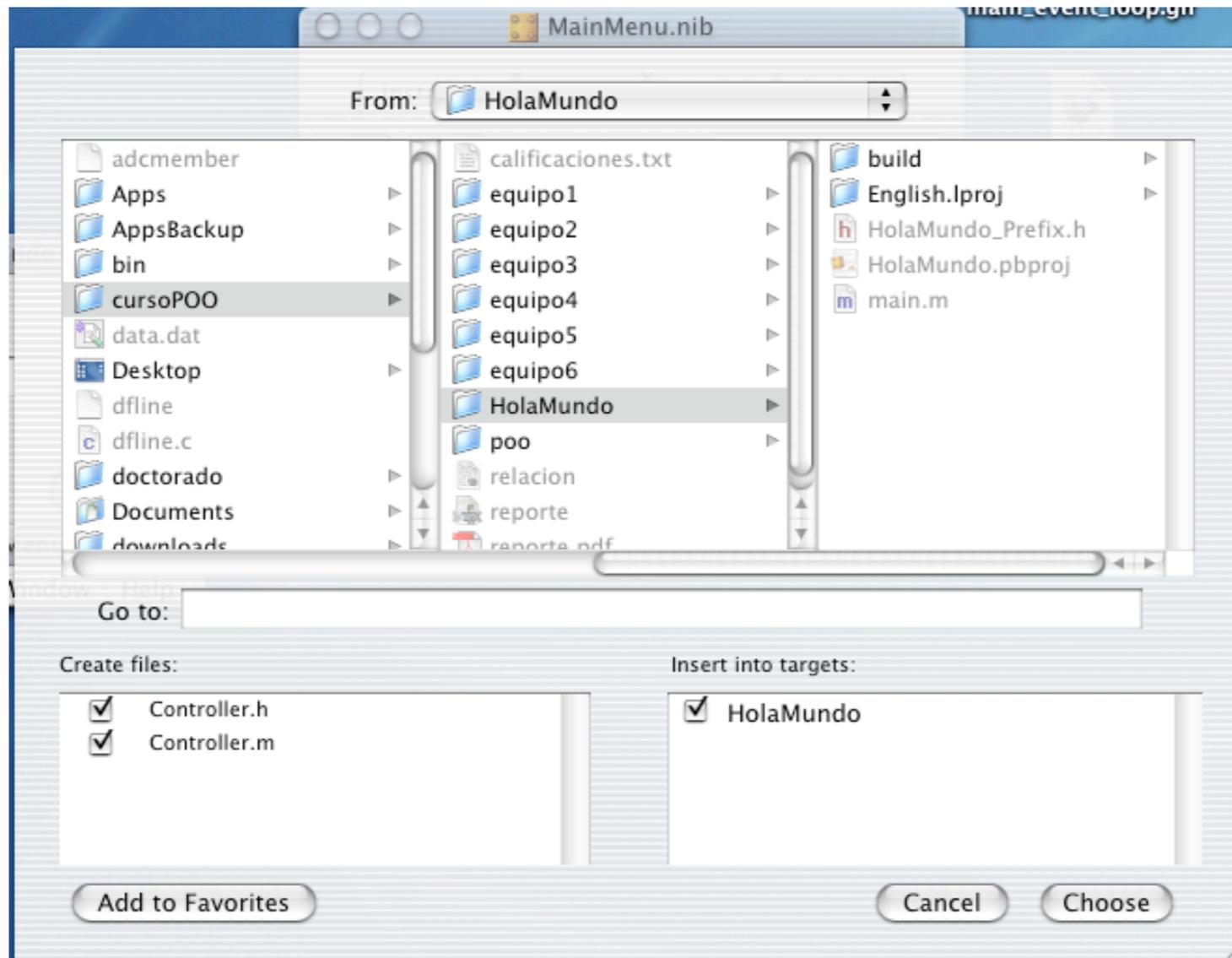
Aplicación *HolaMundo*

- Crear archivos Controller.[hm]



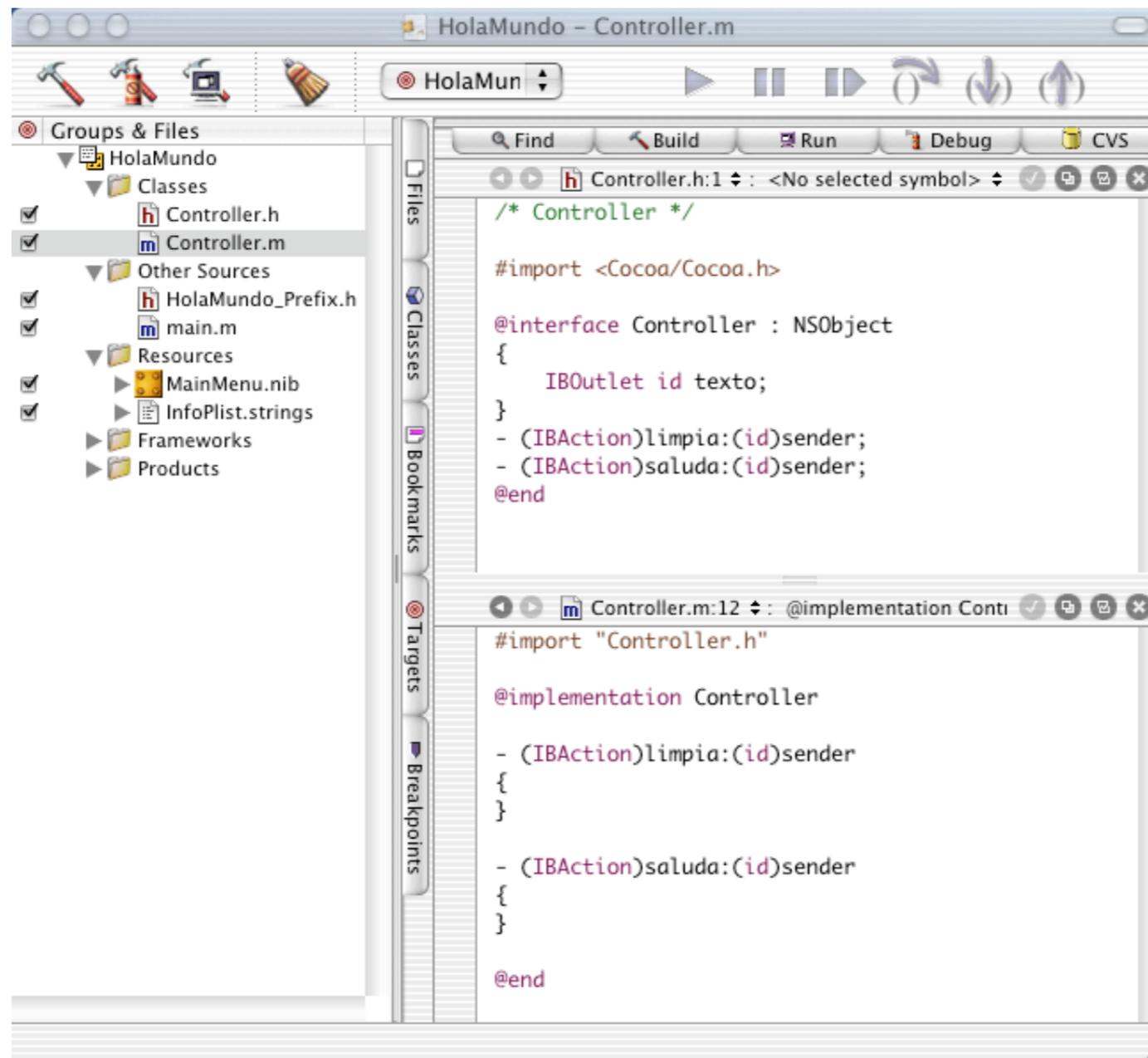
Aplicación *HolaMundo*

- Crear archivos Controller.[hm] (2)



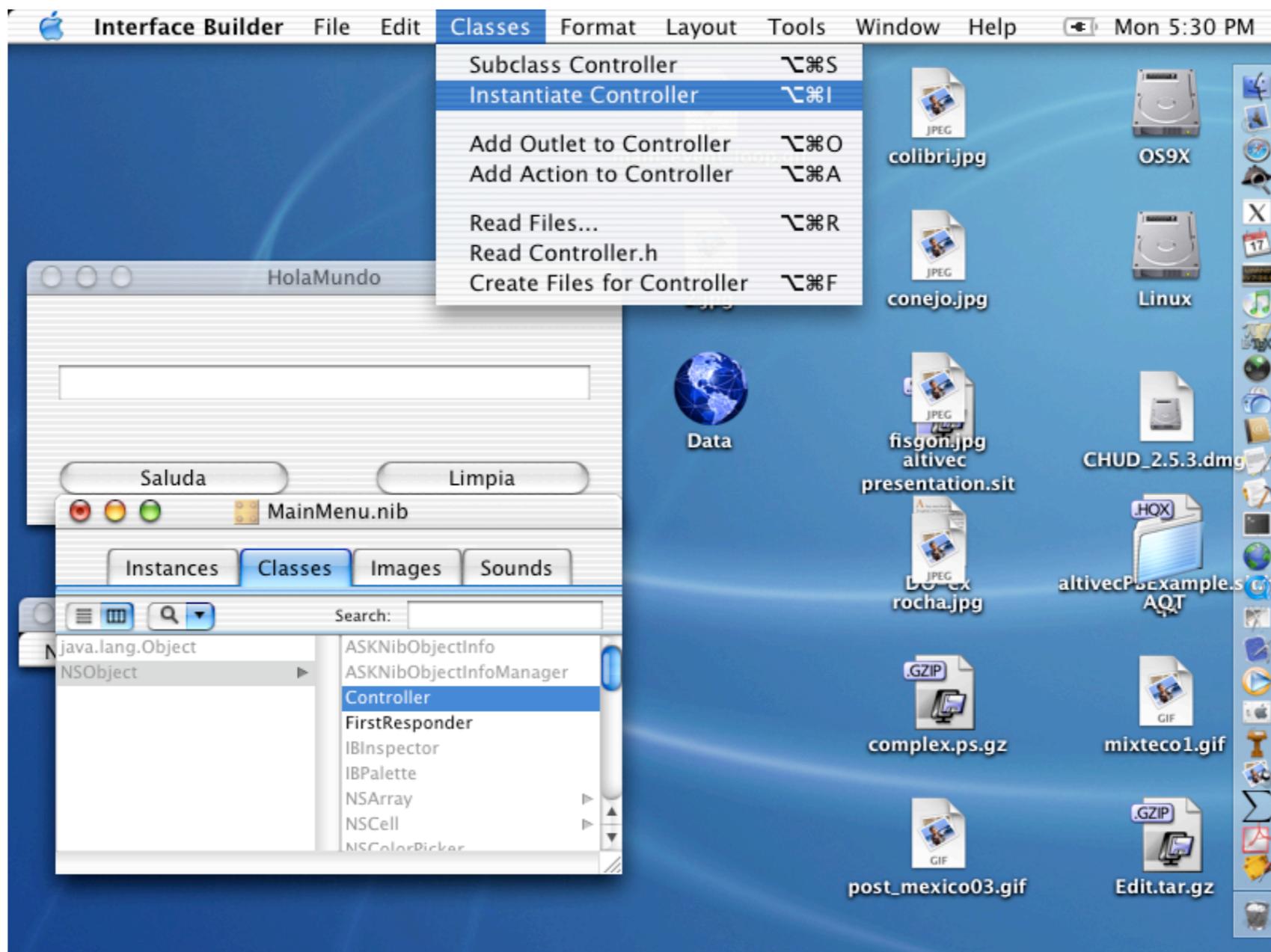
Aplicación *HolaMundo*

- Crear archivos Controller.[hm] (3)



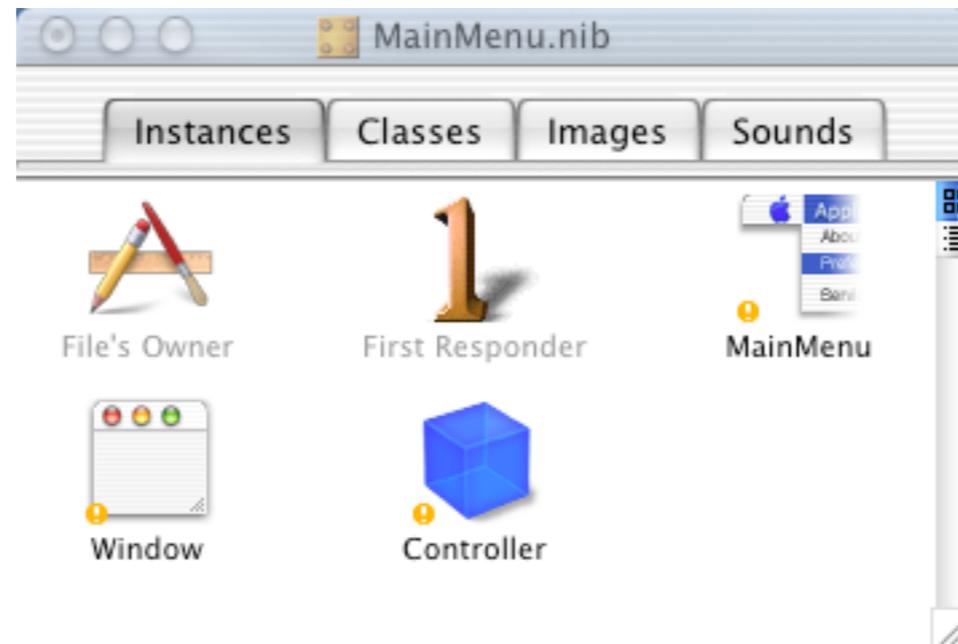
Aplicación *HolaMundo*

- Crear instancia de Controller



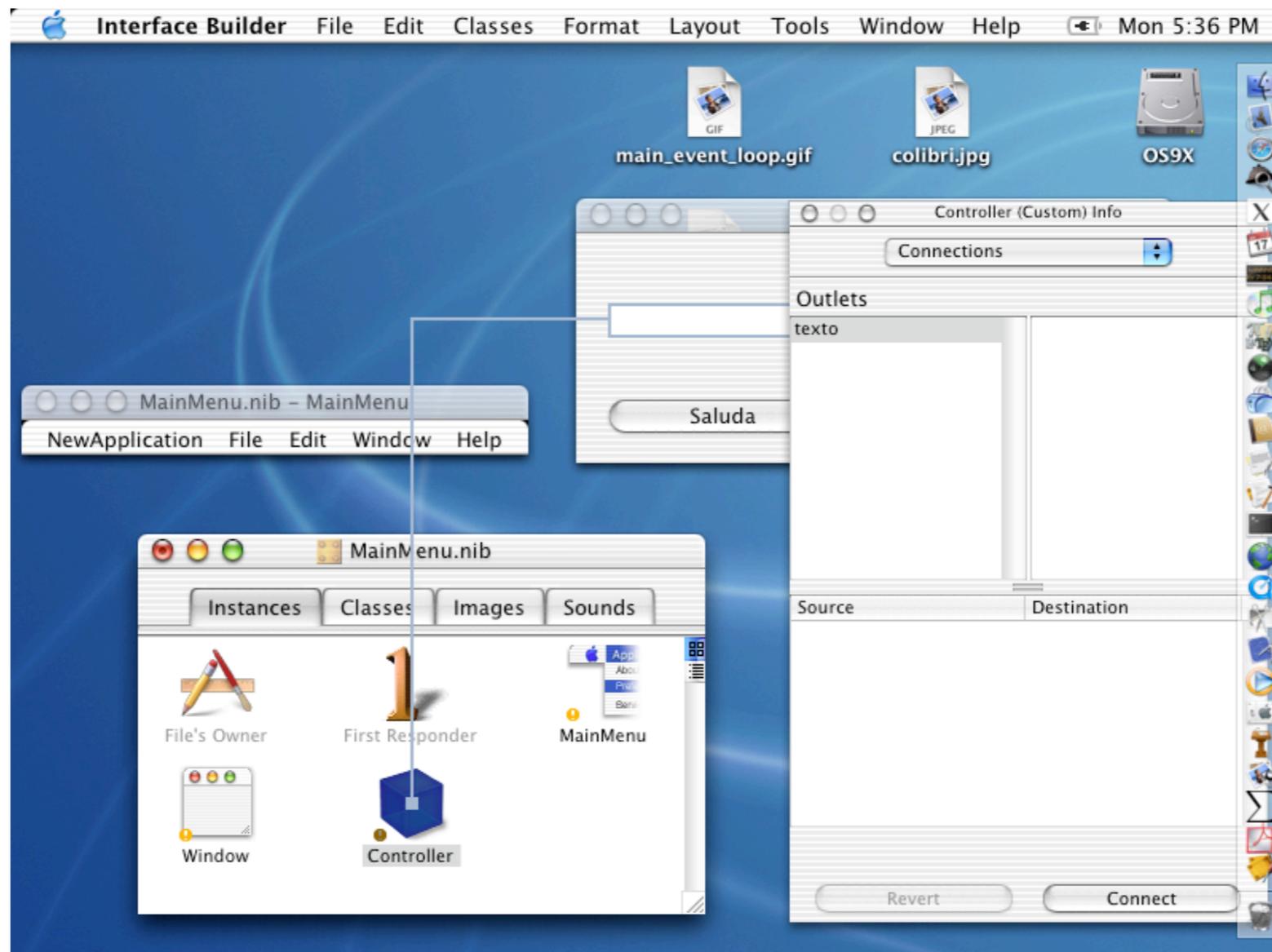
Aplicación *HolaMundo*

- Crear instancia de Controller (2)



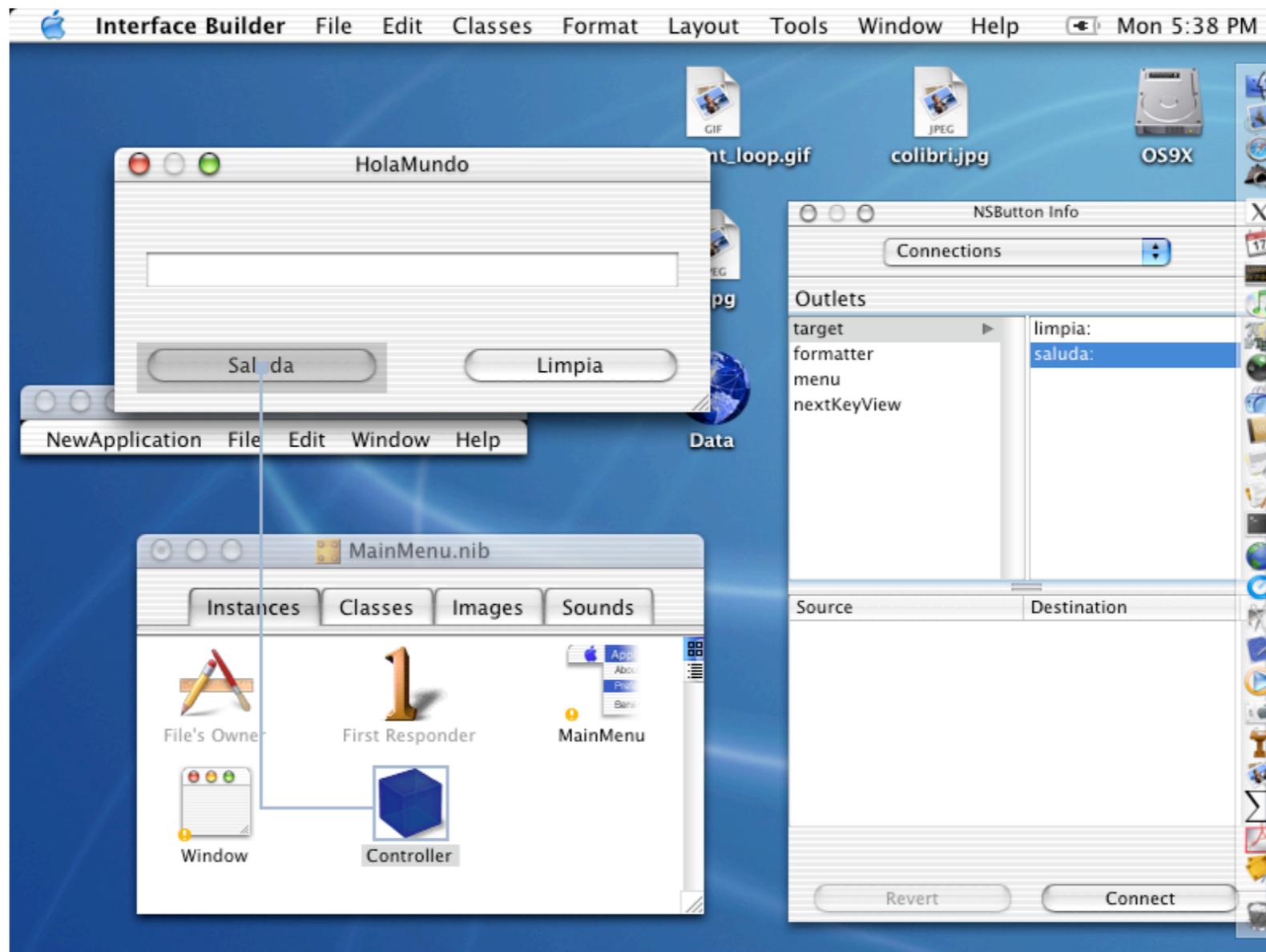
Aplicación *HolaMundo*

- Realizar la conexión al **OUTLET** texto. <CTRL>-Drag desde el objeto controller al objeto texto



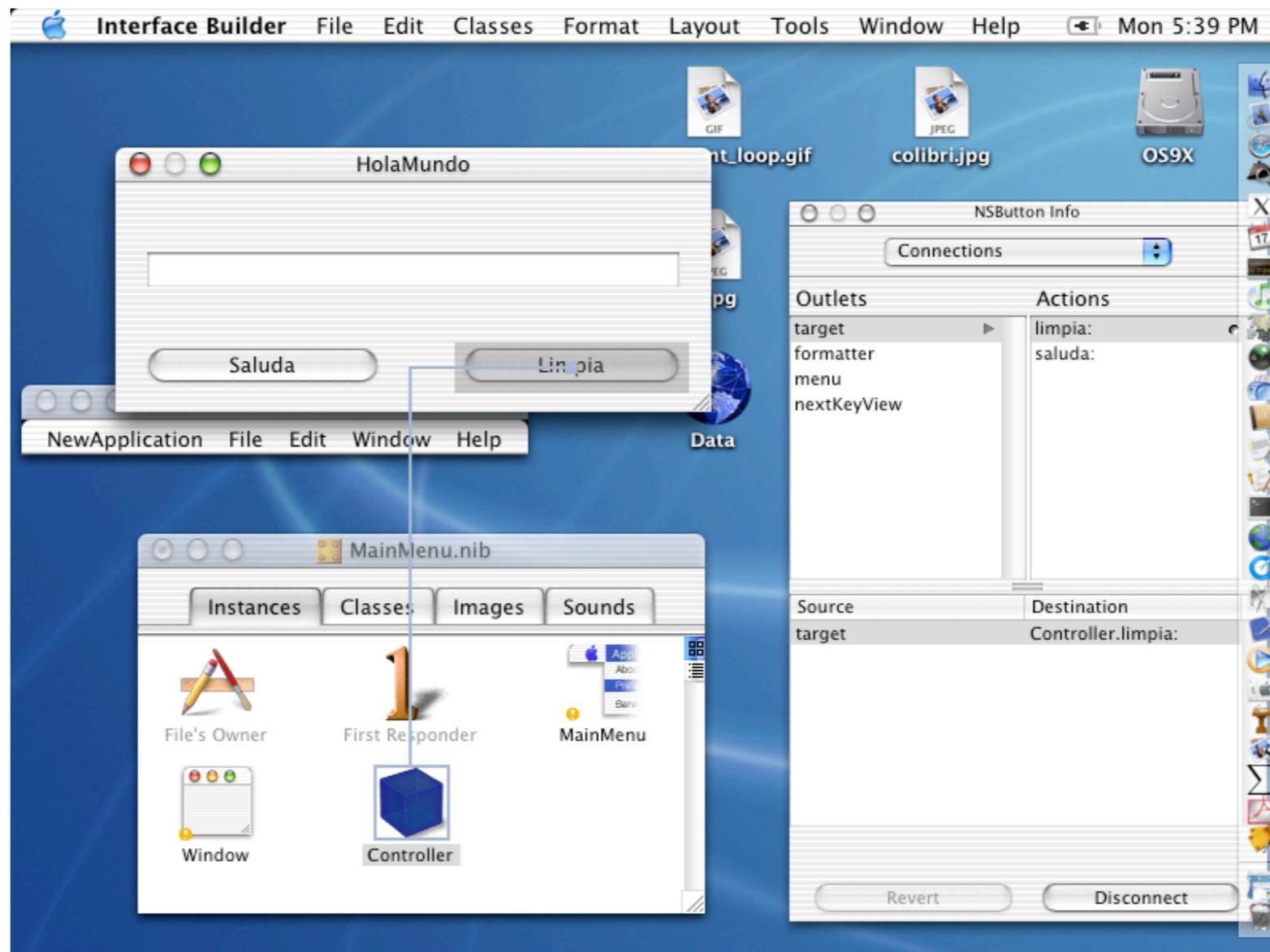
Aplicación *HolaMundo*

- Realizar la conexión a la acción saluda: <CTRL>-Drag desde el objeto botonSaluda al objeto controller



Aplicación *HolaMundo*

- Realizar la conexión a la acción limpia: <CTRL>-Drag desde el objeto botonLimpia al objeto controller



Aplicación *HolaMundo*

- Salvar los cambios hechos en *MainMenu.nib*. Opción Save del submenú File

Aplicación *HolaMundo*

- Modificar el archivo Controller.m

```
#import "Controller.h"

@implementation Controller

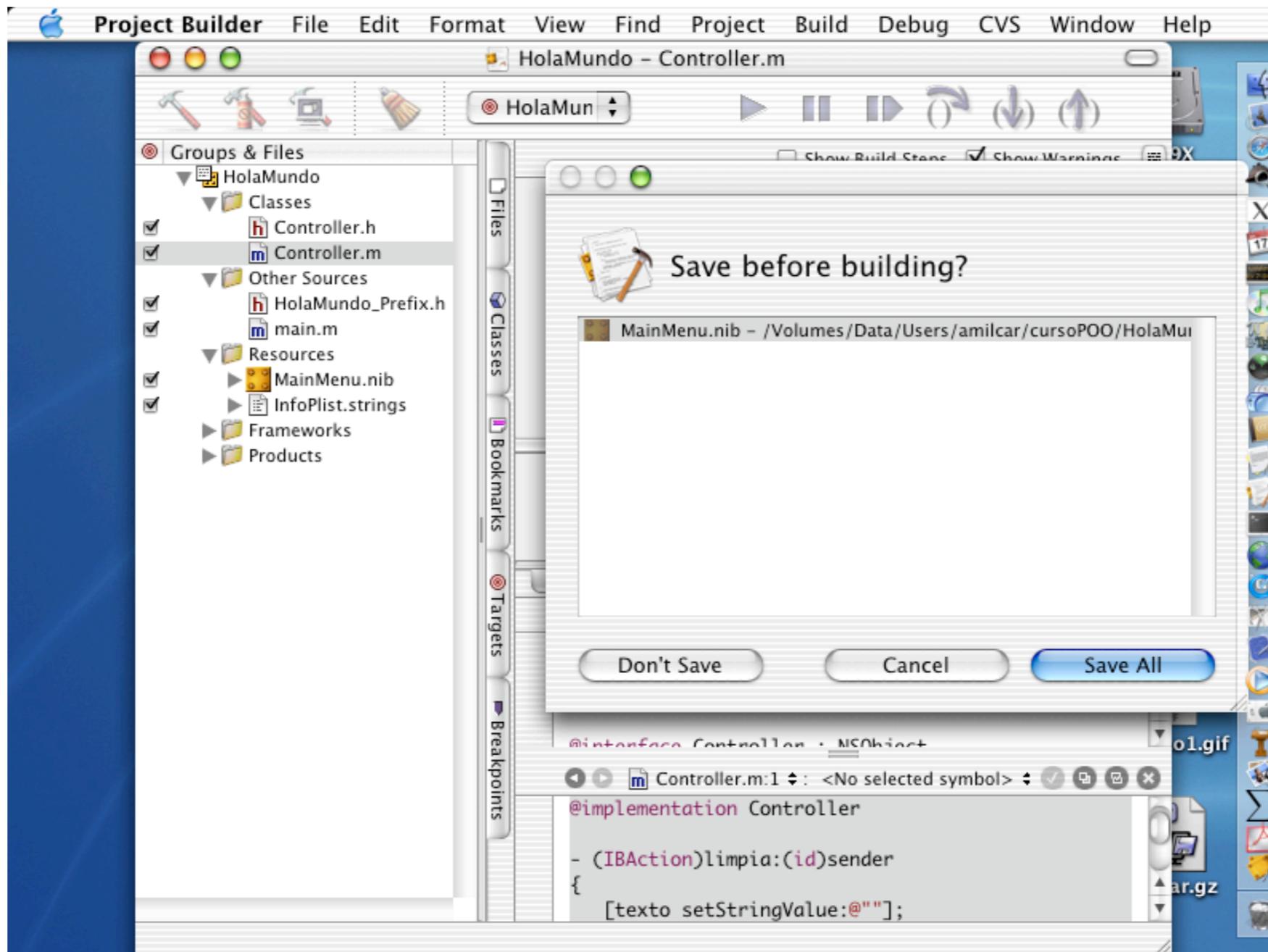
- (IBAction)limpia:(id)sender
{
    [texto setStringValue:@""];
}

- (IBAction)saluda:(id)sender
{
    [texto setStringValue:@"HoLa Mundo"];
}

@end
```

Aplicación *HolaMundo*

- Mandar a construir la aplicación



Aplicación *HolaMundo*

- Probar la aplicación....