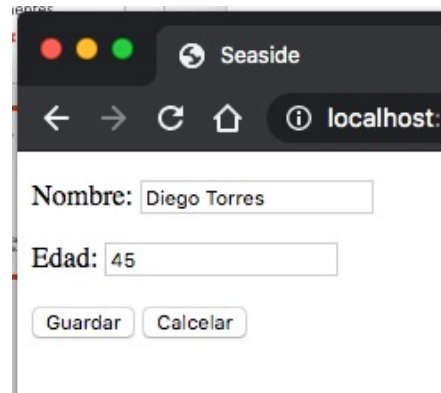
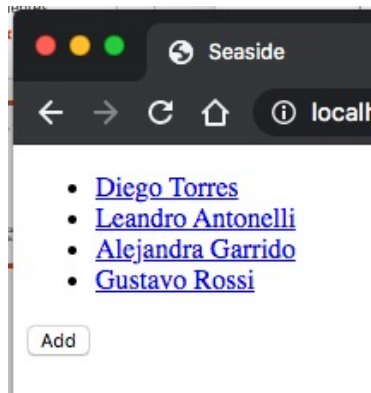


# Formularios y navegación entre componentes

seaside★

# Listar, agregar, mostrar/editar ...



Pinceles que vamos a utilizar:

`unorderedList (<ul>)`, `listItem (<li>)`, `anchor (<a>)`, `paragraph (<p>)`,  
`form (<form>)`, `label (<label>)`, `textInput (<input>)`, `button (<button>)`

# Forms (en general)

- En la web, utilizamos los forms para que el usuario ingrese datos.
- Los forms tienen inputs (textInput, textArea, select, etc.)
- Los inputs se configuran con la información a mostrar
- Los botones envían el formulario y con él los nuevos datos de los inputs



A screenshot of a web browser window. The browser's address bar shows the URL 'Seaside' and 'localhost'. Below the address bar, there is a form with two input fields. The first field is labeled 'Nombre:' and contains the text 'Diego Torres'. The second field is labeled 'Edad:' and contains the number '45'. Below the input fields, there are two buttons: 'Guardar' (Save) and 'Cancelar' (Cancel).

# Forms (Seaside)

- Los inputs necesitan:
  - Obtener el valor para mostrar
  - Guardar el nuevo valor ingresado
- Cuando se oprime un botón, se guardan los nuevos valores de los inputs, y se ejecuta el callback del botón
- El formulario conoce al objeto que edita, y al contenido de los inputs (en variables de instancia)



A screenshot of a web browser window. The title bar shows three colored window control buttons (red, yellow, green) and the text 'Seaside'. The address bar shows navigation icons (back, forward, refresh, home) and the text 'localho'. The main content area displays a form with two text input fields. The first field is labeled 'Nombre:' and contains the text 'Diego Torres'. The second field is labeled 'Edad:' and contains the number '45'. Below the fields are two buttons: 'Guardar' and 'Cancelar'.

- El callback del botón guardar, modifica al objeto editado
- El callback del botón cancelar deja todo como está

# Input (dos formas de configurarlos)

- Con un selector (mensaje) que se usa como getter y setter (en este caso a self, el componente)

```
aCanvas textInput on: #newName of: self
```

- Con #with: y #callback:

```
aCanvas textInput  
    callback: [ :newValue | self newAge: newValue ];  
    with: self newAge ].
```

# Conectando componentes



- El mensaje `#call:`, que entienden los componentes, pasa el control al componente que viene como parámetro (es bloqueante)
- El mensaje `#answer:`, que entienden los componentes, retorna el control al componente que lo llamó