



**UNIVERSIDAD DE SANTIAGO DE CHILE**  
**FACULTAD DE INGENIERÍA**  
**DEPARTAMENTO DE INGENIERÍA INFORMÁTICA**  
**INGENIERÍA CIVIL INFORMÁTICA**



# **MANUAL DE ESTILOS**

## **Interacción Humano Computador**

**G-1 Diurno**

Profesor:  
Edmundo Leiva  
Ayudante:  
Gerardo Mellado  
Jefe de Proyecto:  
Martín Gonzales  
Co-Jefe de Proyecto:  
David Esparza

Santiago de Chile  
2014



# ÍNDICE

1. ESTANDARIZACIÓN DEL ENRUTAMIENTO DE RECURSOS .....	5
2. ESTANDARIZACIÓN DEL DISEÑO DE LAS VISTAS .....	8
3. APLICACIÓN DE HEURÍSTICAS Y PRINCIPIOS .....	11
3.1. APLICACIÓN DE LAS HEURÍSTICAS DE NIELSEN .....	11
3.1.1. Consistencia y estándares .....	11
3.1.2. Libertad y control en manos del usuario .....	11
3.1.3. Minimización de la carga de memoria del usuario .....	11
3.1.4. Prevención de errores.....	12
3.1.5. Flexibilidad y eficiencia.....	12
3.2. APLICACIÓN DE LOS PRINCIPIOS DE LA GESTALT .....	13
3.2.1 Figura - Fondo.....	13
3.2.2 Proximidad y semejanza .....	13
3.2.3 Clausura .....	14
3.2.4 Continuación .....	14



# 1. ESTANDARIZACIÓN DEL ENRUTAMIENTO DE RECURSOS

Para cada recurso se sugiere la utilización de un árbol reflejando las operaciones CRUD originando por cada una de ellas una vista. A continuación, la figura 1.1 refleja el proceso en torno al recurso *photos*.

HTTP Verb	Path	Controller#Action	Used for
GET	/photos	photos#index	display a list of all photos
GET	/photos/new	photos#new	return an HTML form for creating a new photo
POST	/photos	photos#create	create a new photo
GET	/photos/:id	photos#show	display a specific photo
GET	/photos/:id/edit	photos#edit	return an HTML form for editing a photo
PATCH/PUT	/photos/:id	photos#update	update a specific photo
DELETE	/photos/:id	photos#destroy	delete a specific photo

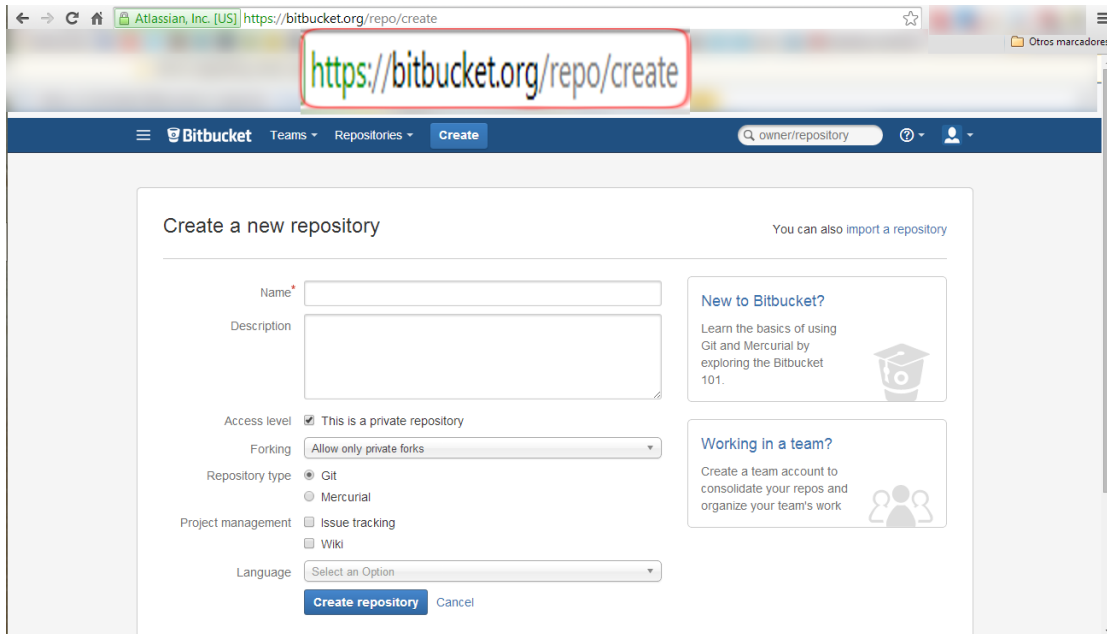
*Figura 1.1. Rutas en torno al recurso photos*

Si bien la tabla se compone de 7 filas, sólo los métodos GET corresponden a vistas. La primera fila responde a la necesidad de listar todas las instancias del recurso *photos*.

La segunda corresponde a la operación CREATE, generalmente estructurada a través de un formulario HTML.

La cuarta fila aplica la operación READ, mientras la quinta UPDATE; nuevamente bajo el esquema de un formulario HTML, y, finalmente, la séptima fila corresponde a la operación DELETE.

Un ejemplo práctico de la aplicación de este principio es apreciable en la página de la implementación de Git: Bitbucket.



*Figura 1.2 Operación CREATE sobre el recurso REPO*

Como se muestra en la figura anterior, basta un vistazo a la ruta del navegador para encontrarse con el estándar: Para el recurso REPO se realiza la operación CREATE.

Ahora bien, esta práctica no es exclusiva de esta compañía, este comportamiento también es identificable en la competencia directa del servicio citado: GitHub.

Completed Set up a personal account

Step 2: Set up the organization

Step 3: Invite team members

### Create an organization

#### Set up the organization

Organization name

The organization will live at <https://github.com/>

Billing email

Receipts will be sent here

#### Choose the organization's plan

SECURE

Plan	Cost	Private repos	
Diamond	\$450/month	300	Choose
Platinum	\$200/month	125	Choose
Gold	\$100/month	50	Choose

#### Organizations

- Repository management
- Fine-grained permissions
- Focused dashboard

The credit card and plan you choose on this screen will be billed to the organization — not your user account

#### Managed by owners

On the next screen you'll be able to grant administrative access to other GitHub users. These people will be able to manage every aspect of the organization (billing, repositories, teams, etc).

*Figura 1.3 Operación CREATE sobre el recurso ORGANIZATION*

En este último ejemplo también se emplea el formulario HTML ya descrito que comúnmente acompaña al proceso de creación de un recurso.

Antes de cerrar este capítulo es necesario explicar cómo el uso de este estándar favorece la usabilidad de una aplicación: Si bien un usuario no asociará los verbos que acompañan al recurso a los métodos HTTP como lo haría un desarrollador, estos resultan suficientemente simples como para describir el propósito de la vista con la que interactúa el usuario, si a esto se suma el apoyo de otras fuentes de información proporcionadas por la misma vista, como *Breadcrumbs*, proveerán una orientación básica favoreciendo el reconocimiento por sobre la memorización a la hora de navegar por la aplicación.

## 2. ESTANDARIZACIÓN DEL DISEÑO DE LAS VISTAS

A continuación se describe detalladamente el diseño propuesto para las vistas del proyecto REVALORA.

En primer lugar, cada vista estará compuesta por un *header* y un *footer*. El *header* contendrá a la izquierda el logo, el nombre de la aplicación y alineado a la derecha el menú principal. Por otro lado, el footer al igual que el header, contendrá al lado izquierdo el logo y el nombre de la aplicación y al costado derecho irán distintos enlaces agrupados por temas, cada uno con su respectivo título. Todo lo mencionado será común a todas las vistas. A continuación se puede apreciar gráficamente lo antes señalado.

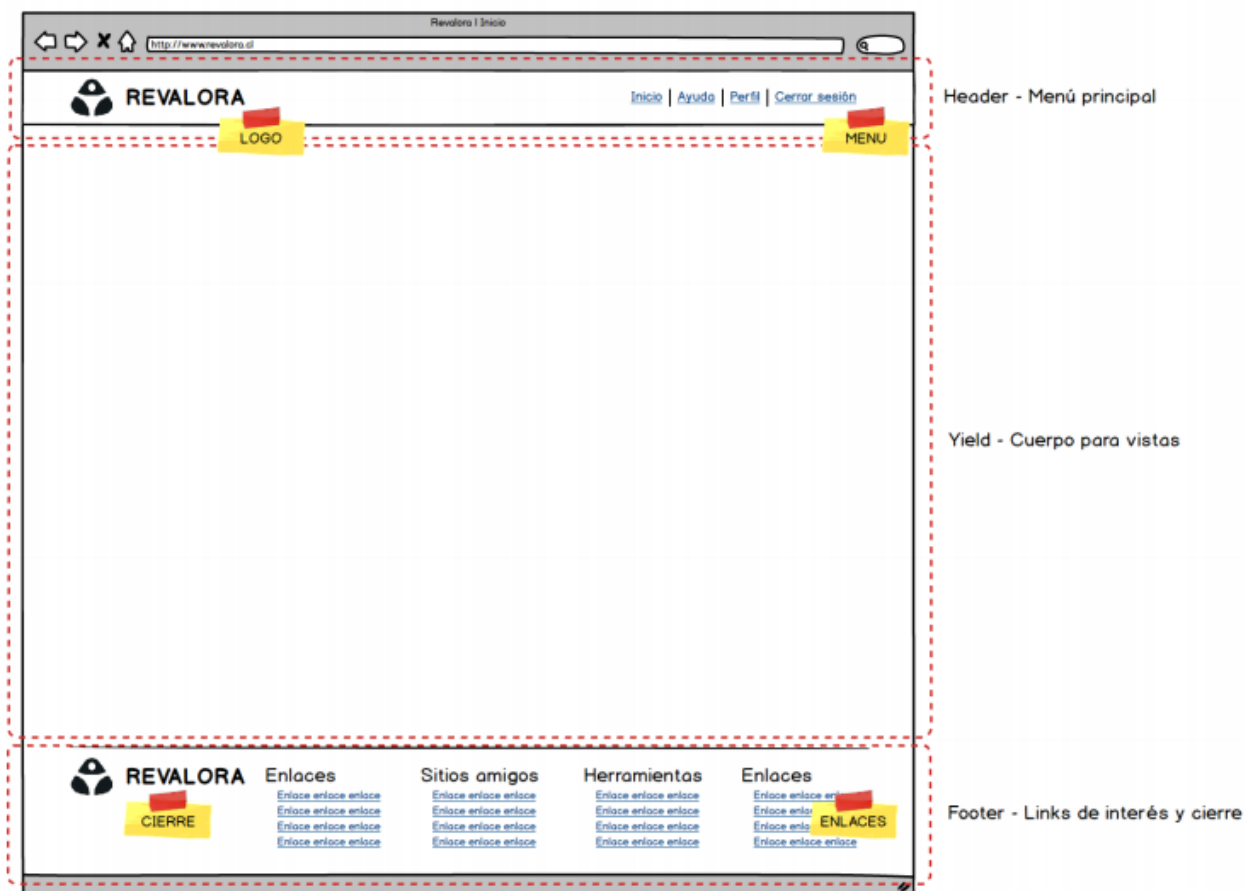


Figura 2.1 "Header y footer"



Como se observó en la imagen anterior, entre el header y el footer se encuentra la *yield*, que corresponde el espacio destinado al contenido a plasmarse en la vista. Pero esta *yield* también seguirá un estándar específico. Cada *yield* contendrá otro header compuesto por un background y un dashboard donde irá el título de la vista. El background permitirá al usuario un mínimo grado de personalización en los proyecto. Siguiendo al header se encuentra el

[Inicio](#) > [Gestion de personas](#) > [Nueva persona](#)

*breadcrumbs* (o las migajas de pan en alusión al cuento de Hansel y Gretel) este elemento representa la ruta de navegación que ha realizado el usuario desde el home hasta la página en la que se encuentra actualmente, como se muestra en la figura 2.2.

Figura 2.2 “Breadcrumbs”

Por último, el contenido que se desea reflejar en la *yield* se ajusta al formato establecido por el *Framework* de *Front-End* de *Twitter: Bootstrap*.

La *yield* será dividida por una rejilla de 12 columnas, por lo tanto si se desea poner una única *div* horizontal, esta tendrá que ocupar las 12 columnas, en resumen, si se quiere tener dos *div* en una fila, la suma de ambas siempre deberá ser igual a un espacio equivalente a las 12 columnas, por ejemplo una podrá ser de 8 y la otra de 4. Finalmente, lo mismo si se desean tener 3 *div* en fila, las tres deberán sumar 12 por lo que la primera *div* podría ser de 3 la segunda de 6 y la última de 3 como se observa en la figura 2.3. Un ejemplo más cercano a lo planteado anteriormente es apreciable en la figura 2.4.

.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1
.col-md-8								.col-md-4			
.col-md-4				.col-md-4				.col-md-4			
.col-md-6						.col-md-6					

Figura 2.3 “Regla de 12”

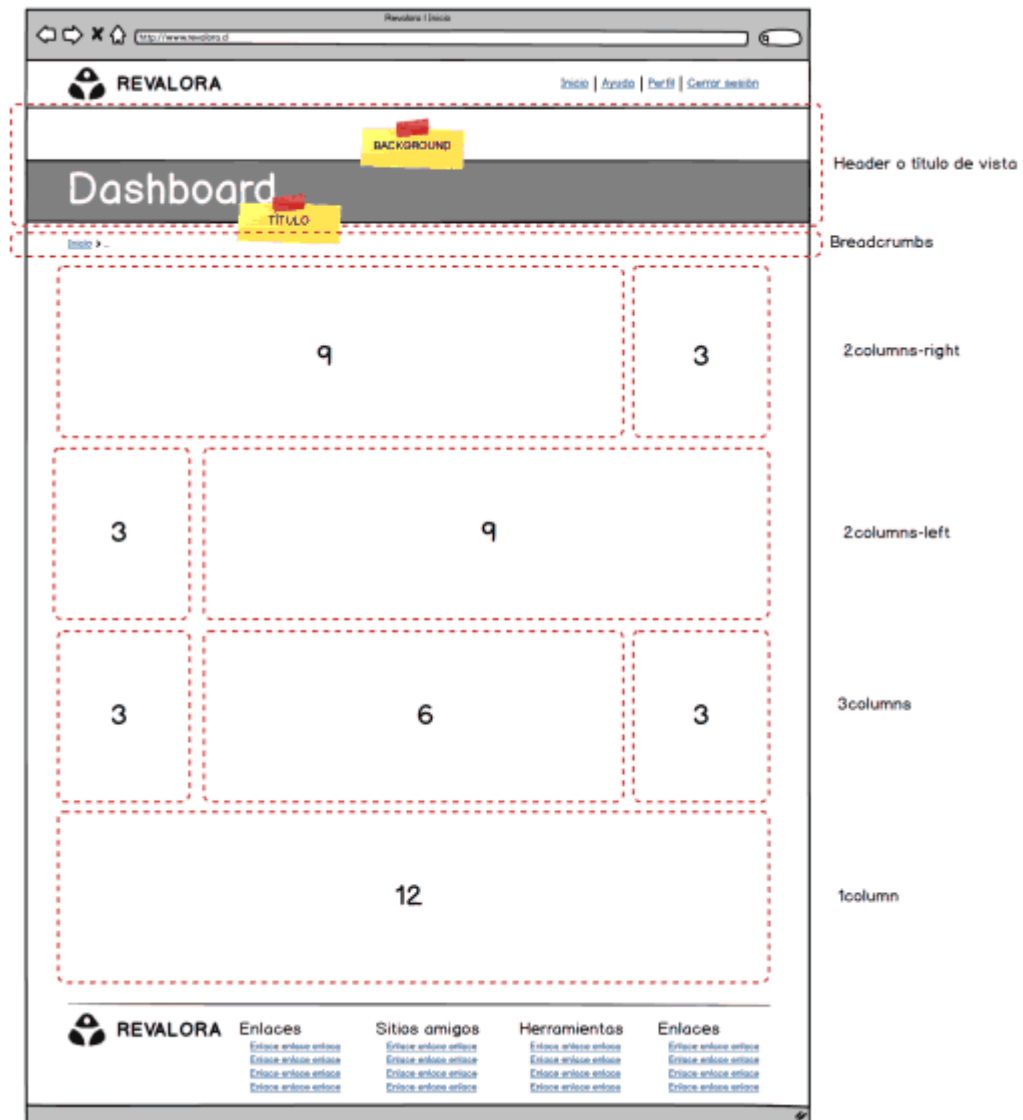


Figura 2.4 “Ejemplo de Vista”

### **3. APLICACIÓN DE HEURÍSTICAS Y PRINCIPIOS**

#### **3.1. APLICACIÓN DE LAS HEURÍSTICAS DE NIELSEN**

En este capítulo se apunta a entregar una lista de prácticas que reflejen la aplicación de las principales heurísticas de Nielsen sobre el sistema REVALORA.

##### **3.1.1. Consistencia y estándares**

Para asegurar la consistencia y el establecimiento de estándares en los capítulos anteriores de este documento se aborda el enrutamiento de recursos, la configuración general de las vistas y la distribución de los contenidos dentro de la *yield* a fin de conservar la homogeneidad en la aplicación.

##### **3.1.2. Libertad y control en manos del usuario**

A través de las *breadcrumbs* o migas de pan se ofrece movilidad al usuario en el sistema, y en general, se propone que el criterio para definir los elementos en cada vista sea la utilidad que puedan ofrecer en relación al módulo en que estas se enmarcan.

##### **3.1.3. Minimización de la carga de memoria del usuario**

En general para evitar la sobrecarga de memoria del usuario se emplean los principios de la Gestalt, como se explicará en secciones posteriores de este capítulo; sin embargo, una restricción crucial es que nunca una vista contenga más de 7 grupos de elementos, conservándose el rango de 5 a 9 elementos acotado por nuestras limitaciones biológicas. Conservar la disposición de las vistas ayuda al usuario a apoyarse en su capacidad de reconocimiento por sobre su memoria, se apunta a este mismo propósito mediante el uso de textos claros e íconos y colores familiares.

### 3.1.4. Prevención de errores

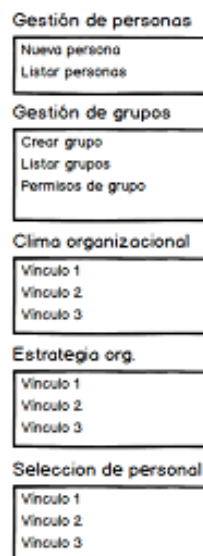
Es necesario reconocer que los usuarios son, como todos, seres humanos proclives a equivocarse, en esta línea, a fin de prevenir que un error implique la ejecución de una operación no deseada se debe hacer uso de ventanas que ofrezcan visibilidad, como se aprecia en la figura 3.1.



*Figura 3.1. Ventana de confirmación de eliminación de un grupo*

### 3.1.5. Flexibilidad y eficiencia

Finalmente, en las vistas correspondientes a índices de cada recurso se ofrece un acceso a las últimas actividades desarrolladas y las principales funciones de cada módulo, aumentando la eficiencia en un marco de flexibilidad, estas opciones se encuentran a la izquierda de la *yield* de dichas vistas.



*Figura 3.2. Acceso directo a trabajos recientes y principales funciones*

## 3.2. APLICACIÓN DE LOS PRINCIPIOS DE LA GESTALT

### 3.2.1 Figura - Fondo

Cuando se desea focalizar la atención del usuario sobre un punto, es recomendable apartarlo del fondo, para ello se usan técnicas como desenfoces y capas de tonos oscuros con opacidades medias, este último caso es el elegido para el sistema a la hora de mostrar una ventana emergente como ilustra la figura 3.2.



Figura 3.3. Aplicación del primer principio de la Gestalt

### 3.2.2 Proximidad y semejanza

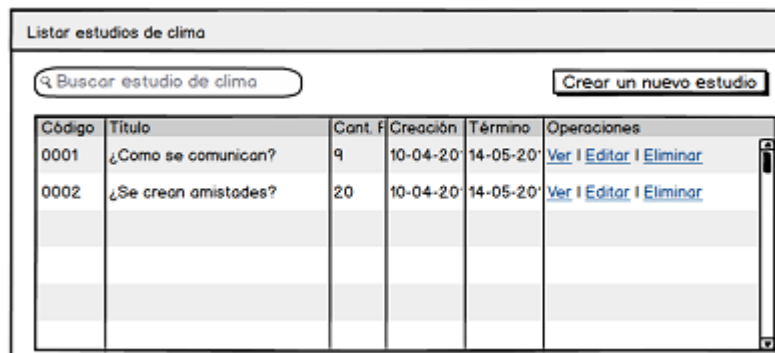
Ambos principios son usados en conjunto para reforzar la idea de grupo a fin de que el usuario considere una colección de elementos con una entidad, por ejemplo los grupos de enlaces ubicados en el *footer* de cada página, que a su vez se pueden reconocer como una macrofamilia de vínculos.



Figura 3.4. Aplicación de los principios de proximidad y semejanza

### 3.2.3 Clausura

Si bien hay espacios claramente definidos por medio de cajas en las vistas, también los hay definidos en exclusiva por este principio como se aprecia en la figura 3.4, allí, la barra de búsqueda y el botón de creación de un nuevo estudio parecieran encerrados en una sola caja donde el límite inferior es dictado por la lista de estudios de clima.



Código	Título	Cant. F	Creación	Término	Operaciones
0001	¿Como se comunican?	9	10-04-20	14-05-20	<a href="#">Ver</a>   <a href="#">Editar</a>   <a href="#">Eliminar</a>
0002	¿Se crean amistades?	20	10-04-20	14-05-20	<a href="#">Ver</a>   <a href="#">Editar</a>   <a href="#">Eliminar</a>

Figura 3.5. Aplicación del principio de clausura

### 3.2.4 Continuación

El principio de continuación tiene una aplicación mucho más simple y es observable en un elemento común a todas las vistas: Las migas de pan. En la aplicación, existen listados y procesos de búsqueda cuyos resultados deben ser mostrados con suficiente claridad al usuario sin que por ello se sature la vista de elementos, Google ofrece una solución práctica empleando el concepto de continuidad que será aplicado en el sistema y se muestra en la figura 3.5.

[< Anterior](#) . [1](#) . [2](#) . [3](#) ... [Siguiente >](#)

Figura 3.6. Aplicación del principio de continuación sobre listas