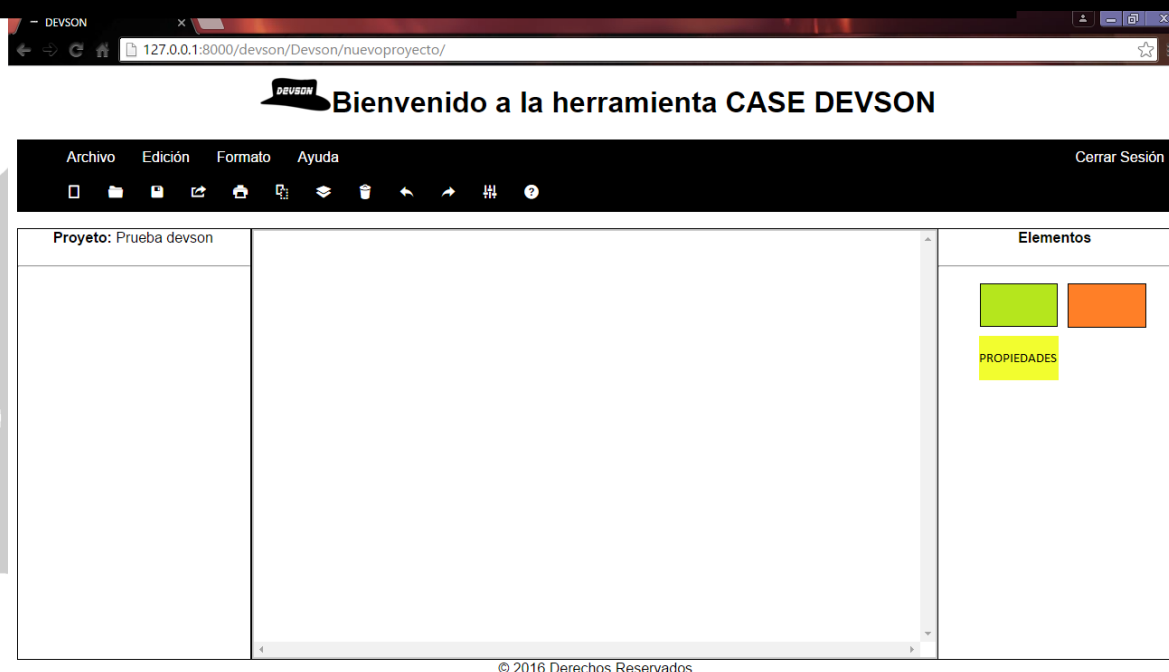




MANUAL DE USUARIO



Daniel Romero

Jhoan Villa

2016

Tabla de contenido

1.	INTRODUCCIÓN	2
2.	CONCEPTOS BÁSICOS	2
2.1.	JSON	2
2.2.	Bases de datos SQL.....	4
2.3.	Bases de datos NoSQL.....	4
3.	FUNCIONES ELEMENTALES.....	5
3.1.	Registrar usuario	6
3.2.	Iniciar sesión.....	8
3.3.	Crear proyecto.....	9
4.	PÁGINA PRINCIPAL	10
4.1.	Barra de menús	11
4.1.1.	Menú archivo	11
4.1.2.	Menú edición.....	12
4.1.3.	Menú formato	13
4.1.4.	Menú ayuda	13
4.1.5.	Cerrar sesión	14
4.2.	Barra de herramientas	14
4.3.	Panel de proyecto	15
4.4.	Área de trabajo.....	15
4.5.	Paleta de elementos.....	16
5.	CREAR UNA ESTRUCTURA DE DATOS.....	16
5.1.	Crear un diagrama.....	16
5.1.1.	Crear un objeto compuesto	17
5.1.2.	Crear un objeto simple	17
5.1.3.	Crear una relación	18
5.2.	Cambiar valor y tipo	20
5.3.	Cambiar aspecto o estilos de objetos	22
5.3.1.	Objeto compuesto y simple	22

5.3.2.	Relación	24
5.3.3.	Texto.....	25
5.3.4.	Ejemplo.....	26
5.4.	Guardar un proyecto	29
5.5.	Abrir un proyecto	30
5.6.	Exportar	32
5.6.1.	Exportar como imagen	32
5.6.2.	Exportar como JSON.....	33
5.6.3.	Exportar como SQL.....	34
5.6.4.	Exportar como Cassandra	34
5.6.5.	Exportar como MongoDB.....	35
5.6.6.	Exportar como Neo4J	35
5.7.	Operaciones complementarias	36
5.7.1.	Copiar objeto.....	36
5.7.2.	Pegar objeto.....	36
5.7.3.	Eliminar objeto.....	37
5.7.4.	Deshacer.....	38
5.7.5.	Rehacer.....	38
REFERENCIAS Y BIBLIOGRAFÍA		39

1. INTRODUCCIÓN

El presente documento describe cuales son las tareas básicas que se pueden ejecutar en la explotación de la herramienta de modelado DEVSON.

El contenido del documento integra, tanto los aspectos de su entorno como características elementales de funcionamiento de la aplicación.

2. CONCEPTOS BÁSICOS

DEVSON es una herramienta CASE para el diseño y construcción de estructuras de datos usadas en los sistemas de software, permitiendo su uso en todas las fases de desarrollo mediante el control de cambios.

DEVSON, permite la realización de ingeniería directa, diseñar y generar estructuras en lenguaje como JSON, SQL y NoSQL (tales como Neo4J, MongoDB y Cassandra) de fácil exportabilidad. El repositorio está basado en DBMS y en repositorios portables, proporcionando buenas respuestas cuando se trata con múltiples usuarios debido a su estructura interna.

2.1. JSON

El lenguaje conocido como JSON es definido como una sintaxis para el almacenamiento e intercambio de datos que requiere uno o más sistemas, esta definición parecida a la que define al XML destaca el hecho de que es una alternativa para el uso de XML. Este lenguaje de marcado fue desarrollado como un complemento del *JavaScript Programming Language* bajo el estándar ECMA-262 (*European Computer Manufacturers Association*) de Diciembre de 1999 (JSON, s.f.).

Su procedencia se debe a JavaScript, esto es porque su nombre completo es *JavaScript Object Notation*. Como XML, este tipo de formato de texto es totalmente independiente al lenguaje de programación en el que se desee implementar, dado esto se dice que JSON es utilizado en toda variedad de lenguajes de programación entre los cuales se ejemplifican C, C++, C#, Java, Python, Perl, JavaScript entre otros.

La sintaxis para escribir JSON se basa en el uso de llaves ({,}), dentro de estas se ingresa la estructura del objeto y cada característica posee un nombre y el valor que este encierra dividido por medio del símbolo de dos puntos (:).

JSON cuenta con una estructura para definir una matriz o “array” contenida por los símbolos de corchetes ([,]) donde cada valor del objeto de esta matriz se separa por una coma (,)(ECMA International).

Dado que JSON se fundó con los desarrolladores de JavaScript es natural que este adopte también los posibles valores básicos que un objeto en JavaScript; puede poseer por ejemplo el de cadenas de caracteres o String, el valor numérico, el arreglo, los valores booleanos tanto verdadero como falso y el valor nulo.

Motores de bases de datos como MongoDB, Neo-4G y Cassandra han dado uso de JSON para almacenar y consultar datos en registros o documentos por medio de una extensión de JSON conocida como BSON o *Binary JSON*, utilizado este método es capaz de generar una eficiente codificación y decodificación de muchos tipos de datos en distintos lenguajes (Colaboradores de MongoDB, s.f.).

Una gran ventaja del uso de esta forma de almacenamiento de JSON es su velocidad y facilidad de acceso de la información sobre todo en lenguajes Scripting, también posee una estructura de datos más liviana que XML gracias a que JSON no requiere incorporar una estructura del objeto específica (DTD), sino que a través de sus marcas permite incluir la descripción de cualquier tipo de objeto incluida su estructura y puede cambiar de forma dinámica.

En la actualidad JSON ha abarcado una gran cantidad de lenguajes importantes de la programación con proyectos, librerías y extensiones; por ejemplo en java se abarca con org.json, Json-lib, Apache johnzon, Json-smart entre otros; en los demás lenguajes de programación como en C++ han destacado JSONKIT, en JavaScript con json2.js u oboe.js sin mencionar a los demás lenguajes de programación.

JSON posee lenguajes de consulta y procesamiento para su propia sintaxis como JSONiq que es un idioma basado en XQuery utilizado para hacer extracción de los datos encerrados en JSON a cualquier otra fuente de datos como lo son las bases de datos; entre las aplicaciones que han usado a JSONiq destacan las encargadas de utilizar información de una base de datos para una plataforma web así como la selección y transformación de datos JSON a XHTML.

La sintaxis JSON usa una colección definida como una secuencia de objetos identificados por medio de un nombre. Las colecciones son usadas para los procesadores de consulta por su capacidad de contener la información (Robie, y otros).

A continuación se puede ver una colección llamada Collection ("país") que contendrá información de algunos países.

```
{
  "país": "Estados unidos",
  "continente": "América",
  "vecinos": ["Canadá", "México"]
}
{
  "país": "Canadá",
  "continente": "América",
  "vecinos": ["Estados unidos", "Alaska"]
}
```

Lo que se desea saber es qué vecinos de Estados Unidos ya están identificados en la colección.

```
For $EstadosUnidos in collection("país"),
  $vecino in collection("país")
Where $EstadosUnidos.pais eq "Estados Unidos"
```

```

And
(some $país in $ EstadosUnidos.vecinos[]
Satisfies $vecino.name eq $país)
return $vecino

```

La consulta anterior encerró en la variable \$EstadosUnidos y \$vecino como todos los países registrados buscando uno por uno hasta hallar aquel que posee el nombre de Estados Unidos para la variable \$Estadosunidos, para luego buscar en todos los registros de países que encierra \$vecino alguno cuyo nombre estuviera en la lista de vecinos de Estados Unidos (Robie, y otros, 2013).

2.2. Bases de datos SQL

SQL es definido de acuerdo con Martin Escofet como un “lenguaje estándar ANSI/ISO de definición, manipulación y control de bases de datos relacionales. Es un lenguaje declarativo” (Escofet).

Este lenguaje tiene definido el uso del modelo relacional para la estructuración de los datos, esto quiere decir que hace uso de tablas. Haciendo una analogía con el modelo entidad relación las tablas serán las entidades, las columnas serán los atributos y las filas serán las tuplas.

2.3. Bases de datos NoSQL

Según Hansel Gracia, Osmel Yanes, Luis Artavia y Mario Villalobos; el término NoSQL (*Not Only SQL*) identifica todas las bases de datos que no siguen completamente el típico modelo relacional y para generar consultas en ellas no suelen utilizar SQL (Gracia del Busto & Yanes Enríquez, 2012) (Artavia & Villalobos).

Estos sistemas de bases de datos se han desarrollado alrededor de empresas como lo son Google, Facebook, Twitter entre otras. La razón por la que estas y otras grandes compañías las usan se debe a que solucionan problemas como el del escalonamiento vertical y procesamiento en tiempo real y en memoria (Acenswhitepapers, 2014).

Al analizar las características de las bases de datos NoSQL se consideran las siguientes (Artavia & Villalobos):

- Tolerancia a fallos y redundancia.
- Estructura distribuida: mejora la distribución de los datos entre las diferentes estructuras o nodos que se manejen.
- Consistencia eventual: Permita facilitar la comunicación entre las estructuras al existir cambios en su contenido.
- Escalabilidad horizontal: Se implementa en nodos de poca capacidad con el fin de utilizar servidores o centros de procesamiento de menos costo.

Dentro de la definición de las bases de datos NoSQL se pueden destacar los siguientes tipos (Gracia del Busto & Yanes Enríquez, 2012):

- Basadas en clave/valor: Este tipo es el que genera el mayor rendimiento en las aplicaciones debido a su sencillez y al hecho que almacena la información (valores) asociándola a unas claves.
- Basadas en documentos: Son similares al tipo clave/valor pero en este caso los valores suelen ser documentos, este tipo se usa para facilitar las consultas complejas.
- Basadas en columnas: A diferencia de las bases de datos relacionales, este tipo almacena los datos en columnas y no en filas, permitiendo la gestión de los datos cuando son compuestos o agregados.
- Basadas en grafos: En este tipo las relaciones se considera y se manejan como un dato que posee la base de datos.
- Basadas en objetos:

Otras: En este grupo se encuentran las bases de datos basadas en tuplas, multi-valuadas, jerárquicas, entre otras que manejan un uso específico y sus implementaciones son escasas.

3. FUNCIONES ELEMENTALES

Los pasos mínimos que debe ejecutar el usuario para utilizar la aplicación se pueden resumir como:

- Registrar sus datos para crear una cuenta de usuario.
- Ingresar a la aplicación con la cuenta de usuario.
- Crear el proyecto.

La presentación inicial de la aplicación consta de un formulario en donde se ingresa el nombre de usuario y la contraseña si ya se realizó el registro con anterioridad, adicionalmente posee las opciones de registrar una nueva cuenta, el acceso al manual de usuario y datos de los creadores (Ver Figura 1).

DEVSON

Bienvenido a la herramienta CASE DEVSON

Registro Manual de usuario Acerca de los desarrolladores

Inicie la sesión aquí

Nombre:

Contraseña:

¿No posee una cuenta? [Regístrese aquí](#)

© 2016 Derechos Reservados.

Figura 1. Página inicial de la aplicación

3.1. Registrar usuario

Para crear una cuenta necesaria para utilizar las funcionalidades de la aplicación, se requiere el diligenciamiento de un formulario que se visualiza al ingresar desde la página inicial (Ver Figura 1) por medio del botón “Regitro” ó por medio del enlace “Regístrese aquí” situado en la parte inferior del formulario de ingreso (Ver Figura 2).



The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/devson/register/". The page features the DEVSON logo, a welcome message "Bienvenido a la herramienta CASE DEVSON", and a "Volver" button. Below this, a heading reads "Complete los siguientes datos para registrarse". The form contains four input fields: "Nombre Completo:" with placeholder "Ingrese su nombre completo", "Nombre:" with placeholder "Ingrese su usuario", "Contraseña:" with placeholder "Ingrese su clave", and "Vuelva a escribir su contraseña:" with placeholder "Confirme su clave". A copyright notice "© 2016 Derechos Reservados." is visible at the bottom.

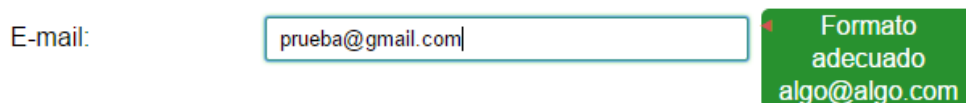
Figura 2. Formulario de registro.

En el formulario se verifican los requisitos de la información, un caso de ejemplo es el ingreso correcto de un e-mail o correo electrónico donde se analiza la estructura propia de este dato (Ver Figura 3 y Figura 4).



The screenshot shows the "E-mail:" label next to an input field containing the text "prueba". To the right of the input field is a red rectangular box with a green checkmark icon and the text "Formato adecuado algo@algo.com", indicating that the current input is not a valid email format.

Figura 3. Ingreso de e-mail.



The screenshot shows the "E-mail:" label next to an input field containing the text "prueba@gmail.com". To the right of the input field is a green rectangular box with a green checkmark icon and the text "Formato adecuado algo@algo.com", indicating that the current input is a valid email format.

Figura 4. E-mail valido.

Para guardar los datos en la parte inferior del formulario se encuentra un botón “Guardar Registro” (Ver Figura 5), si se presenta algún inconveniente como: diferencias en las contraseñas, el nombre de usuario ya existe, el e-mail es invalido o se presenta la ausencia de algún dato; el formulario presentará el mensaje “Verifique sus datos” (Ver Figura 6).

Guardar Registro

¿Ya está registrado? [Inicia Sesión](#)

Figura 5. Botón guardar registro.

Volver

Complete los siguientes datos para registrarse

Verifique sus datos

Nombre Completo:

Nombre:

Contraseña:

© 2016 Derechos Reservados.

Figura 6. Error en el formulario.

Si los datos son correctos el formulario se reemplazará por un mensaje de confirmación y un enlace para regresar a la página inicial (Ver Figura 7).

DEVSON

Bienvenido a la herramienta CASE DEVSON

Volver

Cuenta creada con éxito

[Inicia Sesión](#)

© 2016 Derechos Reservados.

Figura 7. Registro exitoso.

3.2. Iniciar sesión

Para el inicio de sesión, en la página inicial del aplicativo se presenta un formulario que solicita el nombre de usuario y la contraseña que se han establecido al realizar el registro; si el nombre de usuario no existe o la contraseña es errónea se visualizará un mensaje diciendo “Usuario o la contraseña no coinciden” (Ver Figura 8 y Figura 9).

Figura 8. Formulario de ingreso.

Figura 9. Datos erróneos.

Si los datos son correctos, será direccionado a la página que le permitirá crear un proyecto (Ver Figura 10).

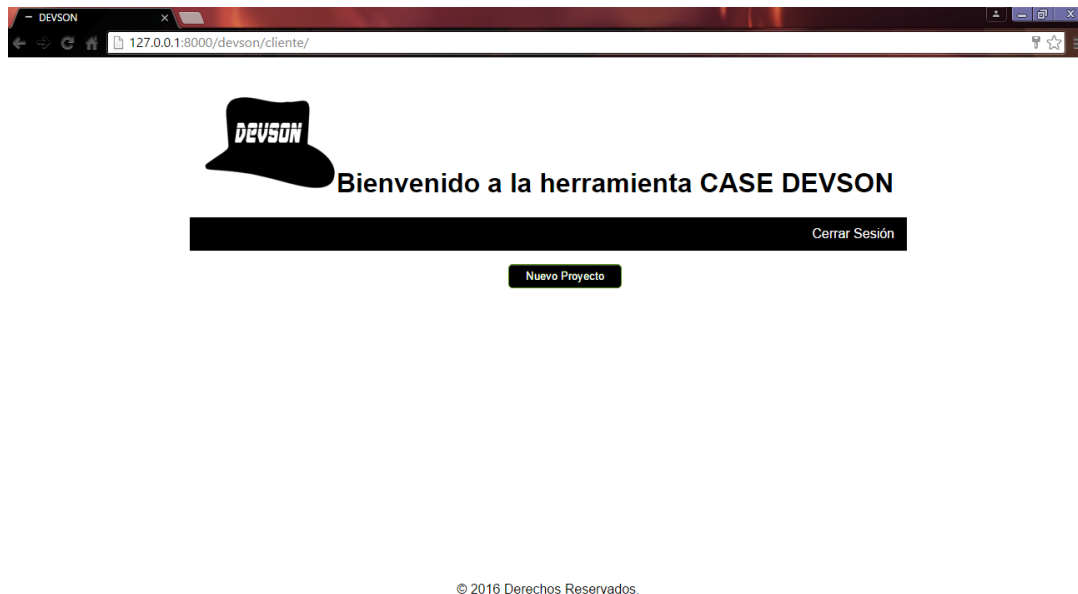


Figura 10. Página para nuevo proyecto.

3.3. Crear proyecto

Una vez el usuario se encuentre en la página del cliente (una vez iniciada sesión) como se visualiza en la Figura 10, al presionarse el botón “Nuevo Proyecto” (Ver Figura 11) se mostrará un mensaje emergente del navegador que solicita el nombre que va a poseer el nuevo proyecto (Ver Figura 12).

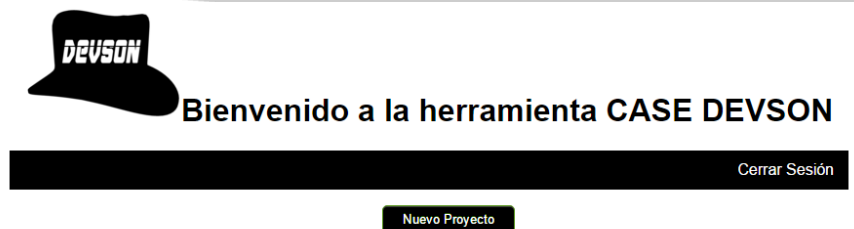


Figura 11. Botón de nuevo proyecto.

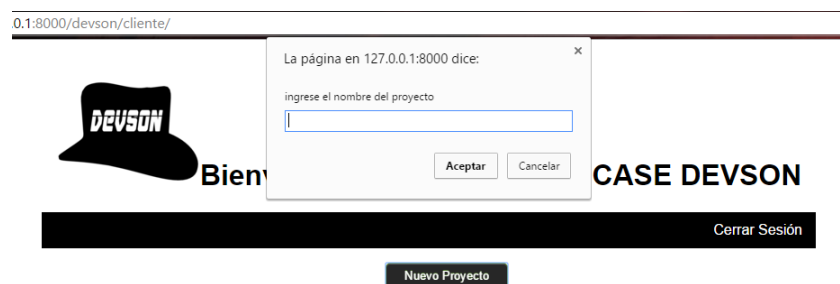


Figura 12. Mensaje emergente.

Una vez se ingrese el nombre del proyecto y se presione el botón aceptar (Ver Figura 13), se direccionará a la página principal de la aplicación en donde se podrá desarrollar el modelado; se podrá visualizar que en la parte izquierda de la página se encontrará el nombre del proyecto indicando que fue creado exitosamente (Ver Figura 14).

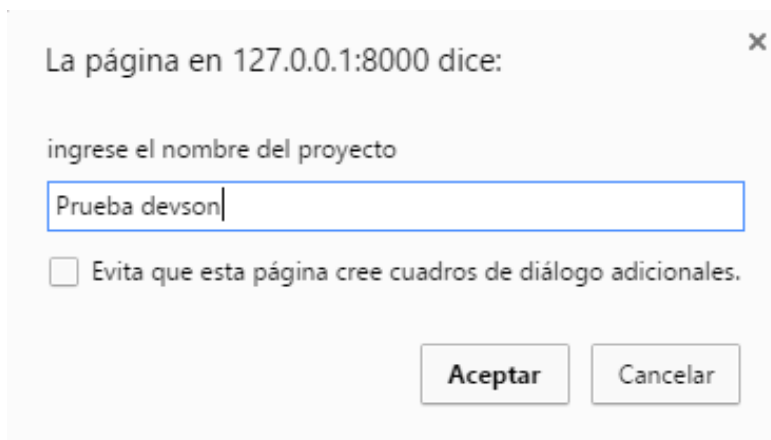


Figura 13. Ingreso del nombre del proyecto.

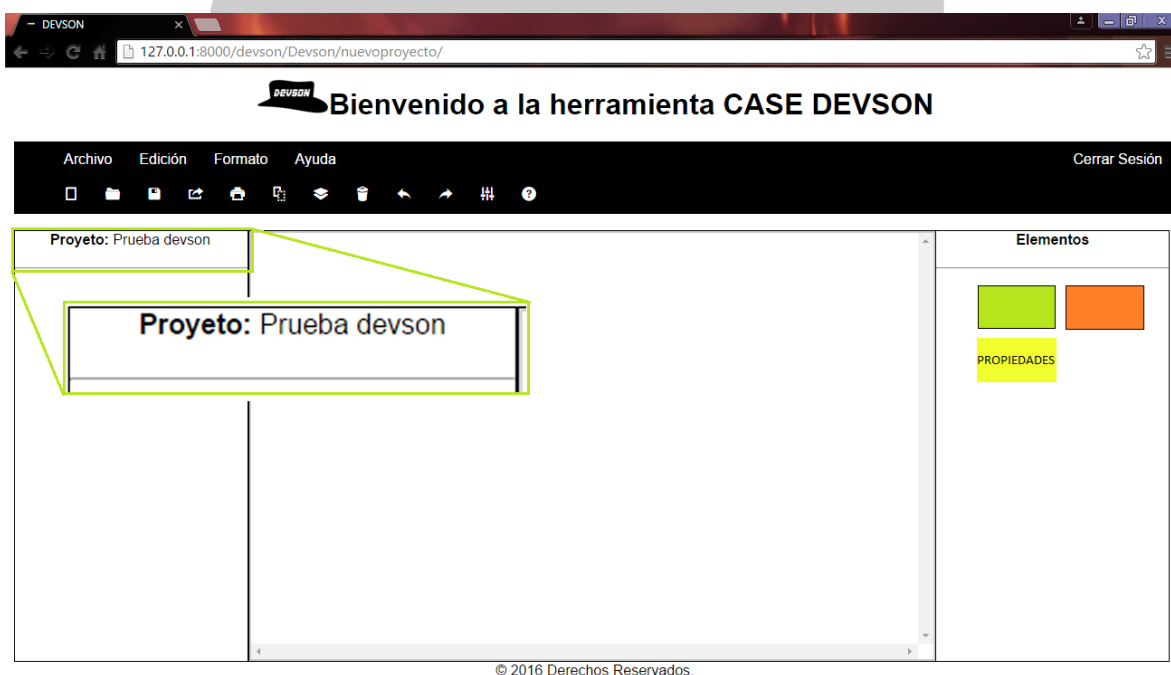


Figura 14. Página de desarrollo del proyecto.

4. PÁGINA PRINCIPAL

Al crear un nuevo proyecto aparece una página como se visualiza en la Figura 15, la cual se compone de las siguientes partes:

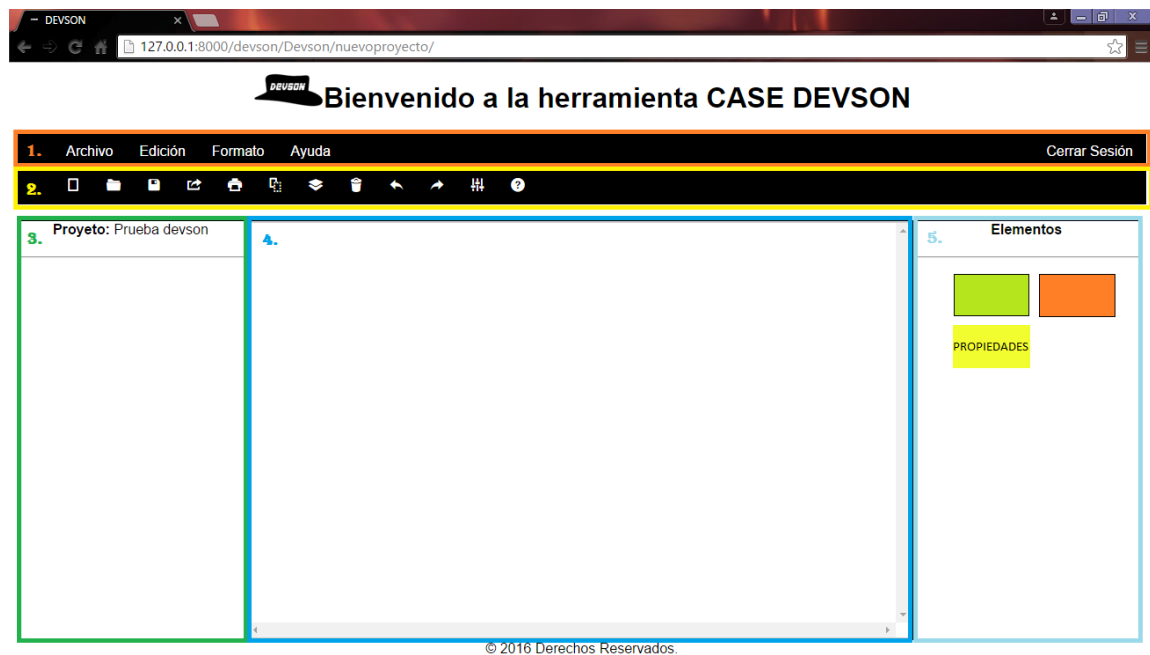


Figura 15. Página principal.

1. Barra de menús.
2. Barra de herramientas.
3. Panel de proyecto.
4. Área de trabajo.
5. Paleta de elementos.

4.1. Barra de menús

Contiene las operaciones de DEVSON, agrupadas en menús desplegables relacionados con los diferentes elementos que maneja la aplicación (Ver Figura 16). Todas las operaciones fundamentales se pueden hacer a partir de estos menús, aunque se realizan más rápidamente a partir de los iconos de la barra de herramientas.



Figura 16. Barra de menús.

4.1.1. Menú archivo

En este menú se encuentran las operaciones asociadas al proyecto que se desarrolla como se ve en la Figura 17, entre las funcionalidades de este menú se encuentran:

- **Nuevo:** Permite crear un nuevo proyecto re-direccionando a la página del cliente (Ver Figura 10).
- **Abrir:** Permite al usuario abrir un proyecto existente desde la base de datos de la aplicación o mediante la importación de un archivo generado con anterioridad.

- **Guardar:** Permite al usuario guardar un proyecto en la base de datos de la aplicación o mediante la exportación de un archivo generado con la estructura del proyecto para que pueda ser abierto desde cualquier cuenta.
- **Exportar como:** Esta opción permite convertir el diagrama generado en el área de trabajo en una estructura de datos en formatos como JSON, SQL y NoSQL; en esta última opción se encuentran disponibles los formatos para Cassandra, MongoDB y Neo4J.

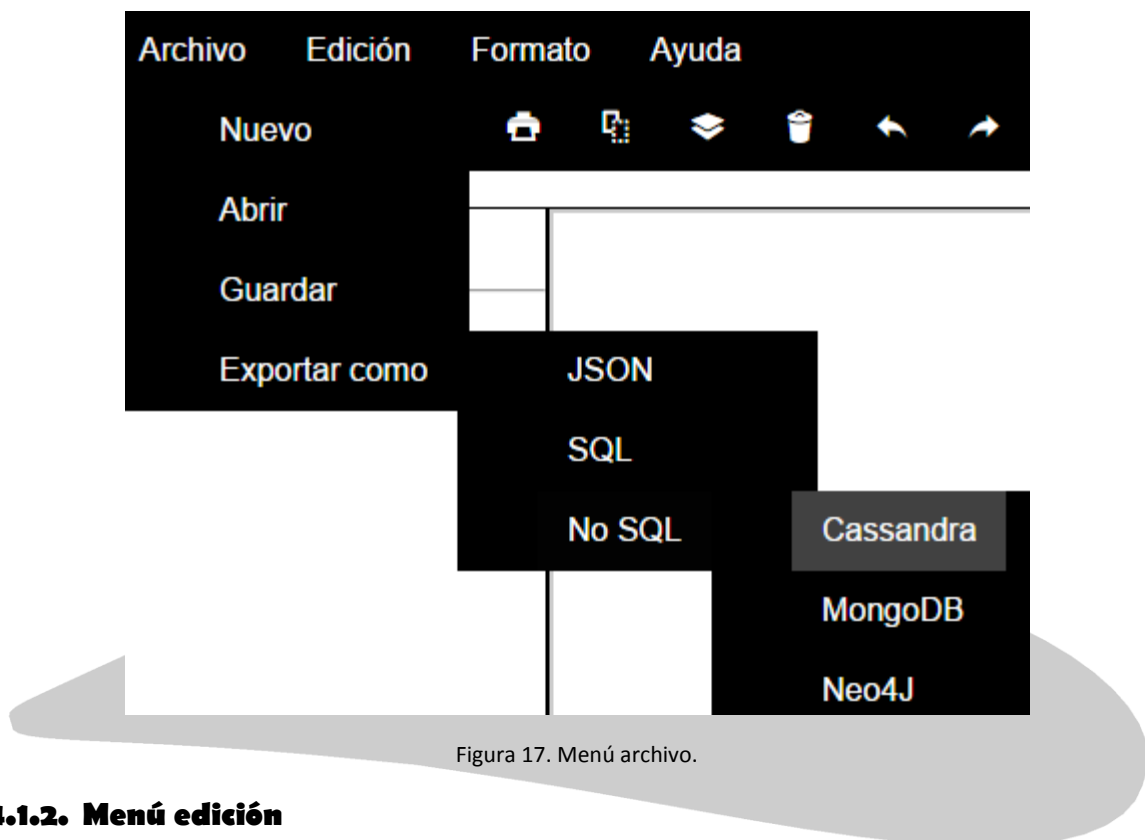


Figura 17. Menú archivo.

4.1.2. Menú edición

En este menú se encuentran las operaciones asociadas a la edición del área de trabajo como se ve en la Figura 18, entre las funcionalidades de este menú se encuentran:

- **Deshacer:** Permite regresar a un diagrama anterior eliminando la última acción realizada.
- **Rehacer:** Permite regresar a un diagrama posterior insertando la última acción eliminada.
- **Copiar:** Permite copiar un elemento del área de trabajo.
- **Pegar:** Permite pegar un elemento copiado en el área de trabajo.

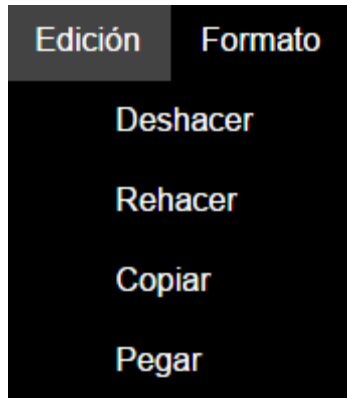


Figura 18. Menú edición.

4.1.3. Menú formato

En este menú se encuentran las operaciones asociadas a la presentación de los elementos creados en el área de trabajo como se ve en la Figura 19, entre las funcionalidades de este menú se encuentran:

- **Objeto simple:** Permite cambiar la presentación de los objetos simples (hojas).
- **Objeto compuesto:** Permite cambiar la presentación de los objetos compuestos (nodos).
- **Relación:** Permite cambiar la presentación de las relaciones existentes entre los elementos creados.
- **Texto:** Permite cambiar la presentación del texto que posee cada elemento.

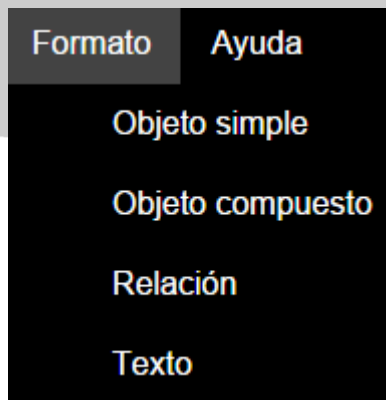


Figura 19. Menú formato.

4.1.4. Menú ayuda

En este menú se encuentran las operaciones que permiten al usuario tener información de la aplicación en caso de ser necesaria como se ve en la Figura 20, entre las funcionalidades de este menú se encuentra la observación del manual de usuario.

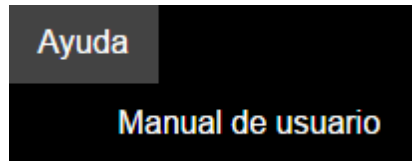


Figura 20. Menú ayuda.

4.1.5. Cerrar sesión

En este menú se encuentran la operación que permite al usuario cerrar la sesión creada y ser re-direccionado a la página inicial de la aplicación (Ver Figura 21)

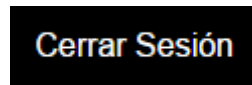


Figura 21. Cerrar sesión.

4.2. Barra de herramientas

Contiene las operaciones de DEVSON, agrupadas en iconos para ejecutar de forma inmediata (Ver Figura 22). Entre los iconos se encuentran las siguientes operaciones:



Figura 22. Barra de herramientas.

- **Nuevo proyecto:** Permite crear un nuevo proyecto re-direccionando a la página del cliente (Ver Figura 10).
- **Abrir proyecto:** Permite al usuario abrir un proyecto existente desde la base de datos de la aplicación o mediante la importación de un archivo generado con anterioridad.
- **Guardar proyecto:** Permite al usuario guardar un proyecto en la base de datos de la aplicación o mediante la exportación de un archivo generado con la estructura del proyecto para que pueda ser abierto desde cualquier cuenta.
- **Exportar como JSON:** Permite convertir el diagrama generado en el área de trabajo en una estructura de datos en formato JSON.
- **Imprimir como imagen:** Permite descargar el diagrama creado como una imagen con formato PNG.
- **Copiar objeto:** Permite copiar un elemento del área de trabajo.
- **Pegar objeto:** Permite pegar un elemento copiado en el área de trabajo.
- **Eliminar objeto:** Permite eliminar un elemento creado en el área de trabajo.
- **Deshacer:** Permite regresar a un diagrama anterior eliminando la última acción realizada.
- **Rehacer:** Permite regresar a un diagrama posterior insertando la última acción eliminada.
- **Cualidades de objetos:** Permite cambiar la presentación de los objetos.
- **Manual de usuario:** Permite la observación del manual de usuario.

4.3. Panel de proyecto

Esta sección se divide en dos partes: en la parte superior se visualiza el nombre del proyecto, el cual es ingresado al momento de crear un nuevo proyecto y en la parte inferior se crea una vista preliminar de la estructura de datos en formato JSON (Ver Figura 23).

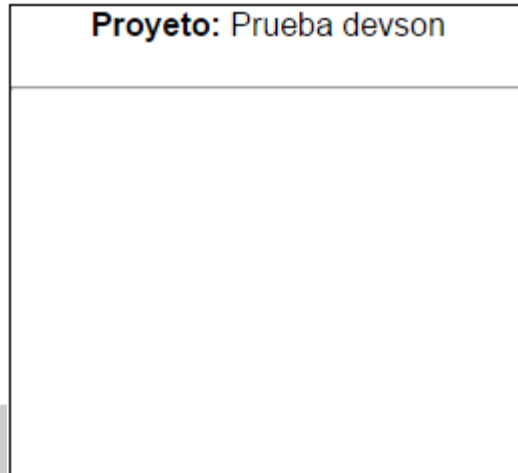


Figura 23. Panel de proyecto.

4.4. Área de trabajo

Esta sección es la encargada de aceptar los objetos que se desean crear para el desarrollo del diagrama y posterior estructura de datos (Ver Figura 24).



Figura 24. Área de trabajo.

4.5. Paleta de elementos

Esta sección se encuentra los iconos que permiten la creación de los objetos para modelar la estructura de datos dentro del área de trabajo (Ver Figura 25), los iconos disponibles son:

- Crear objeto compuesto: Crea un objeto de tipo nodo que puede contener uno o más objeto.
- Crear objeto simple: Crea un objeto de tipo hoja, el cual solo puede tener un tipo y valor pero no se permite que posea otro objeto.
- Propiedades: Permite cambiar el tipo y el valor del objeto seleccionado en el área de trabajo.

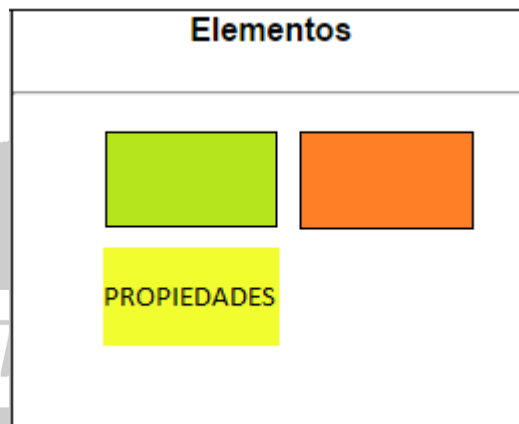


Figura 25. Paleta de elementos.

5. CREAR UNA ESTRUCTURA DE DATOS

Los pasos que puede ejecutar el usuario para la creación de una estructura de datos se pueden resumir en:

- Crear el diagrama.
- Cambiar el valor y el tipo al objeto.
- Cambiar el aspecto o estilo a los objetos.
- Guardar/Abrir el proyecto.
- Exportar.
- Operaciones complementarias.

5.1. Crear un diagrama

Los pasos mínimos que debe ejecutar el usuario para crear el diagrama consisten en:

- Crear un objeto compuesto.
- Crear un objeto simple.
- Crear una relación.

5.1.1. Crear un objeto compuesto

Para crear un objeto compuesto o un objeto que representa un nodo, se presiona en la paleta de elementos el botón “Insertar un objeto compuesto” (Ver Figura 26) y luego hacer clic en cualquier punto dentro del área de trabajo.



Figura 26. Botón insertar un objeto compuesto.

Una vez realizado lo anterior se visualizará el objeto como se muestra en la Figura 27, el cual posee dos botones con los que se podrá agregar o quitar las conexiones para crear las relaciones y una etiqueta que contiene el nombre del objeto (predeterminadamente será object). Al momento de crear el objeto se visualizará en forma de un objeto JSON en el panel de proyecto dando una vista preliminar de la estructura de datos que se crea (Ver Figura 28).

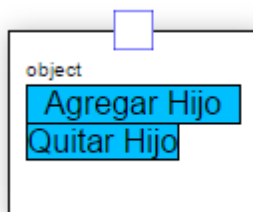


Figura 27. Objeto compuesto creado.

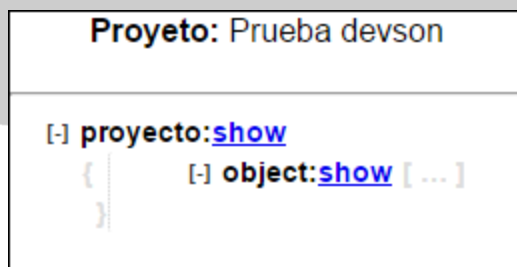


Figura 28. Objeto compuesto en el panel de proyecto.

5.1.2. Crear un objeto simple

Para crear un objeto simple o un objeto que representa una hoja, se presiona en la paleta de elementos el botón “Insertar un objeto simple” (Ver Figura 29) y luego hacer clic en cualquier punto dentro del área de trabajo.



Figura 29. Botón insertar un objeto simple.

Una vez realizado lo anterior se visualizará el objeto como se muestra en la Figura 30, el cual posee una etiqueta que posee el nombre del tipo del objeto que predeterminadamente

no posee (aparecerá no hay tipo). Al momento de crear el objeto se visualizará en forma de un objeto JSON en el panel de proyecto dando una vista preliminar de la estructura de datos que se crea (Ver Figura 31).

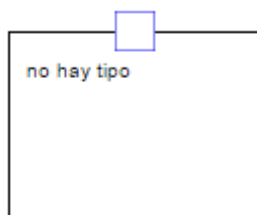


Figura 30. Objeto simple creado.

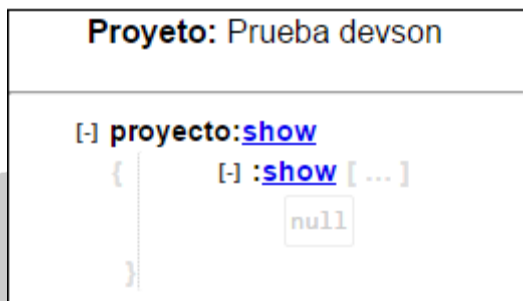


Figura 31. Objeto simple en el panel de proyecto.

5.1.3. Crear una relación

Para conectar dos objetos (por ejemplo los que se visualizan en la Figura 32), se hace un clic sostenido desde la conexión del objeto compuesto (cuadro pequeño ubicado en la parte superior del objeto) hasta la conexión del objeto que se quiere convertir en un hijo; mientras se hace eso se visualizará una flecha direccionándose desde el objeto padre al objeto hijo (ver Figura 33).

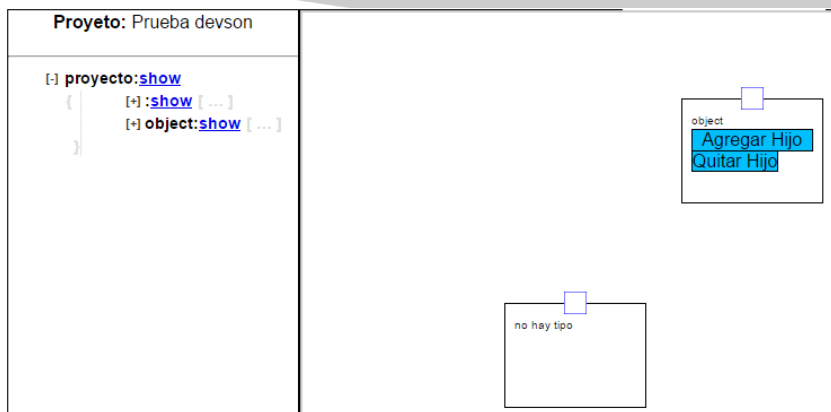


Figura 32. Objetos creados.

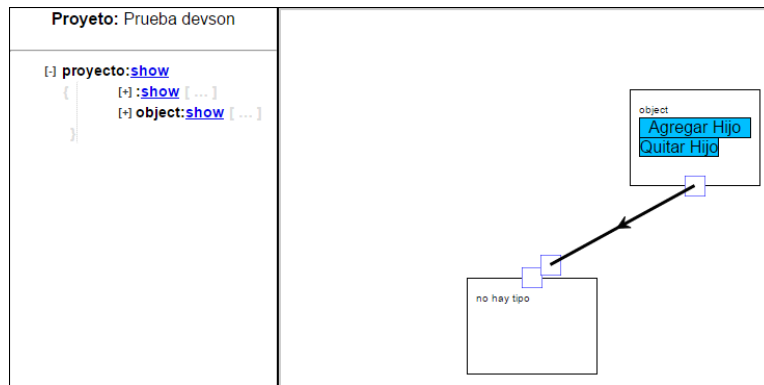


Figura 33. Creación de una relación.

Una vez termine de realizar el clic sostenido se observa la conexión terminada entre los elementos y en el panel del proyecto el objeto hijo será trasladado al interior del objeto padre (ver Figura 34).

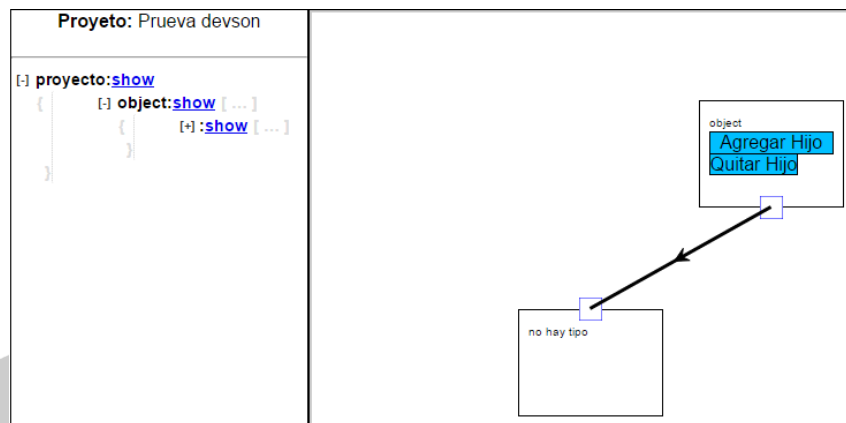


Figura 34. Relación terminada.

Se debe considerar que un objeto compuesto puede ser ingresado como padre o como hijo, a diferencia de un objeto simple que solo puede tener una relación de hijo (ver Figura 35).

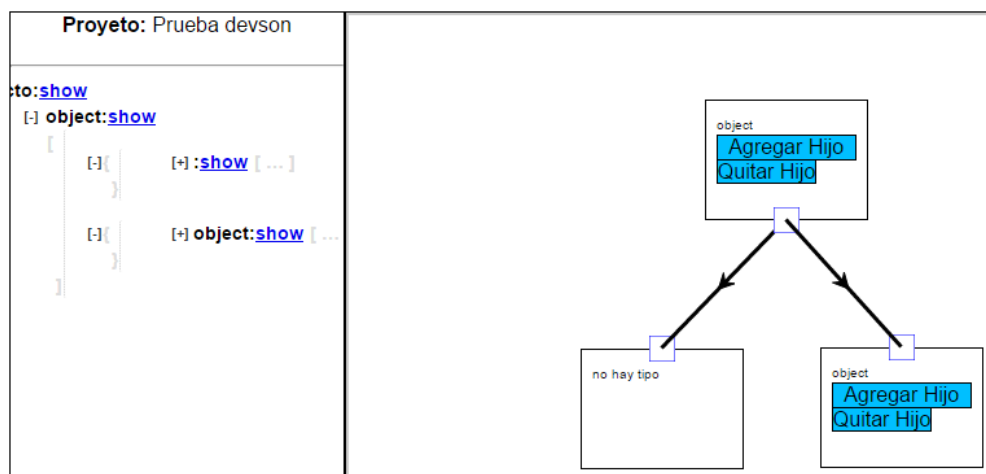


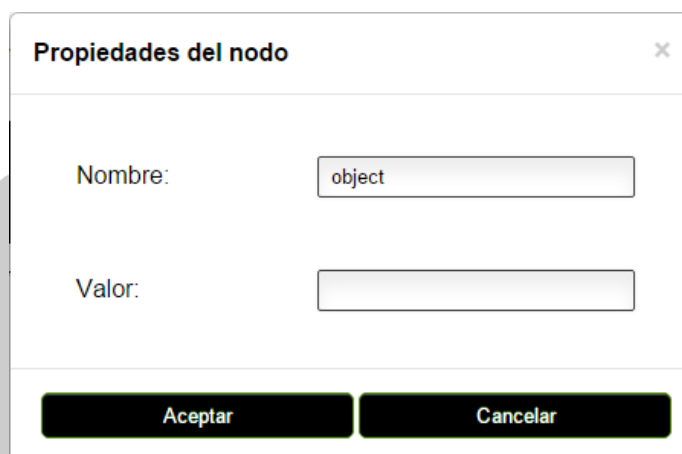
Figura 35. Objeto compuesto hijo.

5.2. Cambiar valor y tipo

Para cambiar el valor y el tipo (nombre) a un objeto creado en el área de trabajo se presiona el botón “Cambiar las propiedades de un nodo” (ver Figura 36) y luego hacer clic sobre el elemento a modificar; se abrirá un mensaje emergente con un formulario que solicitará los datos de nombre y valor. Si se trata de un objeto compuesto se mostrará como nombre “object” y el campo de valor se encontrará vacío (ver Figura 37).

PROPIEDADES

Figura 36. Botón cambiar las propiedades de un nodo.



Propiedades del nodo

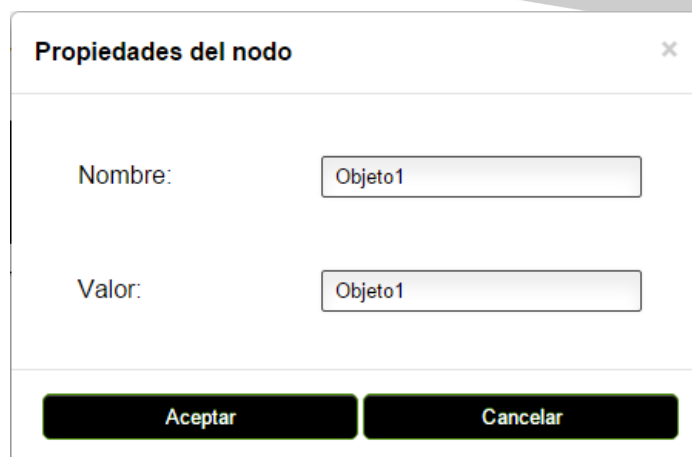
Nombre: object

Valor:

Aceptar Cancelar

Figura 37. Formulario para un objeto compuesto.

Luego de ingresar los datos dentro de los campos (por ejemplo como se visualiza en la Figura 38) y presionar el botón “Aceptar” el nodo escogido cambiará el nombre dentro de su etiqueta y en la estructura JSON en el panel de proyecto (ver Figura 39).



Propiedades del nodo

Nombre: Objeto1

Valor: Objeto1

Aceptar Cancelar

Figura 38. Ingreso de datos.

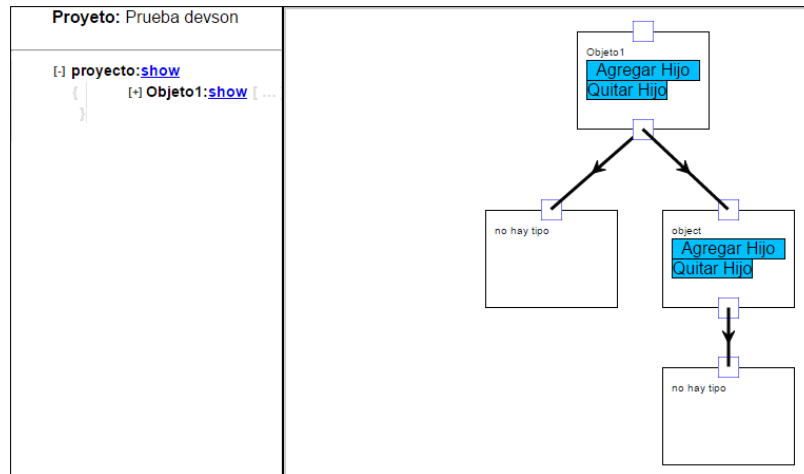


Figura 39. Objeto compuesto modificado.

Si se trata de un objeto simple se mostrarán los campos nombre y valor vacíos; luego de ingresar los datos dentro de los campos (por ejemplo como se visualiza en la Figura 40) y presionar el botón “Aceptar” el nodo escogido cambiará el nombre dentro de su etiqueta y en la estructura JSON en el panel de proyecto (ver Figura 41).

Figura 40. Ingreso de datos.

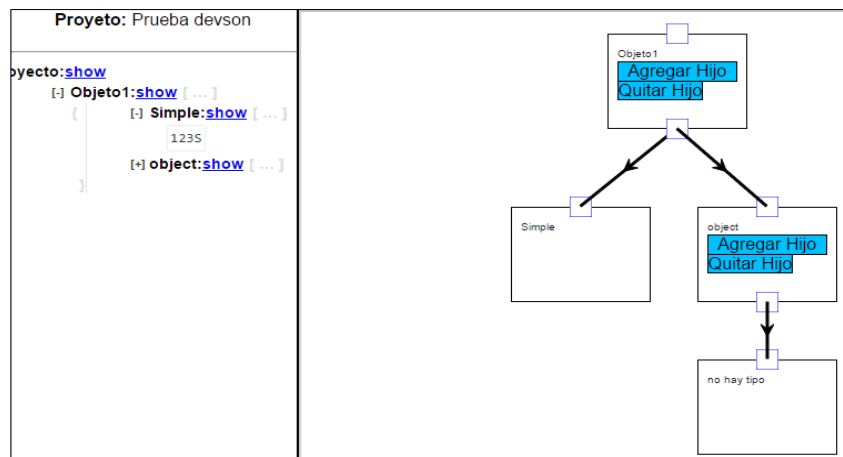



Figura 41. Objeto simple modificado.

5.3. Cambiar aspecto o estilo de objetos

Para cambiar la presentación o el estilo de un tipo de elemento del área de trabajo, se accede desde el menú Formato y se escoge la opción requerida ó desde el botón “Cualidades de objetos” .

Las propiedades disponibles para cambiar varían según el tipo de elemento deseado como:

- Objeto Compuesto y Simple
- Relación
- Texto

5.3.1. Objeto compuesto y simple

Al escoger la opción de “Objeto simple” o “Objeto compuesto” se abre una ventana emergente con los campos Grosor de línea, Color de línea y Color (Color de relleno) como se observan en la Figura 42 y Figura 43.



Propiedades - Google Chrome

127.0.0.1:8000/devson/propiedades/

Objeto Compuesto Objeto Simple Relación Texto

Grosor Línea: 0

Color Línea: 

Color: 



Guardar Cambios

Figura 42. Cualidades objeto compuesto.

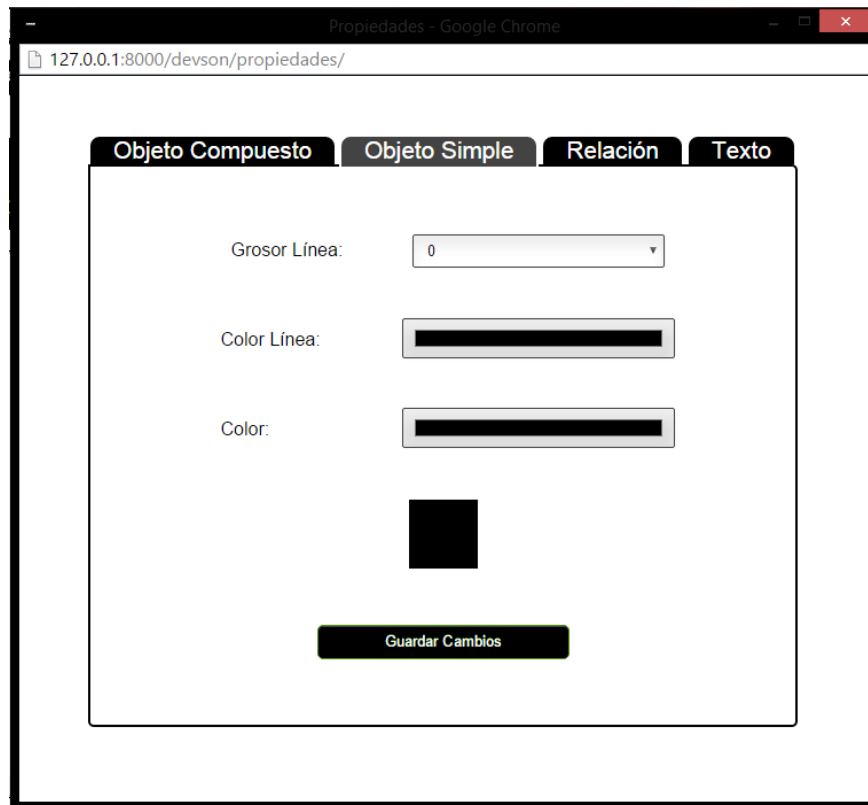


Figura 43. Cualidades objeto simple.

En el campo Grosor de línea se despliega una lista con tamaños de grosor que van desde 0 hasta 10 píxeles (ver Figura 44), en los campos de Color de línea y Color se presiona el botón que se encuentra al lado de cada etiqueta y aparecerá un selector de color para elegir de una gran variedad de tonos (ver Figura 45); los cambios se visualizan en el objeto de ejemplo dentro de la ventana.



Figura 44. Selección de grosor.

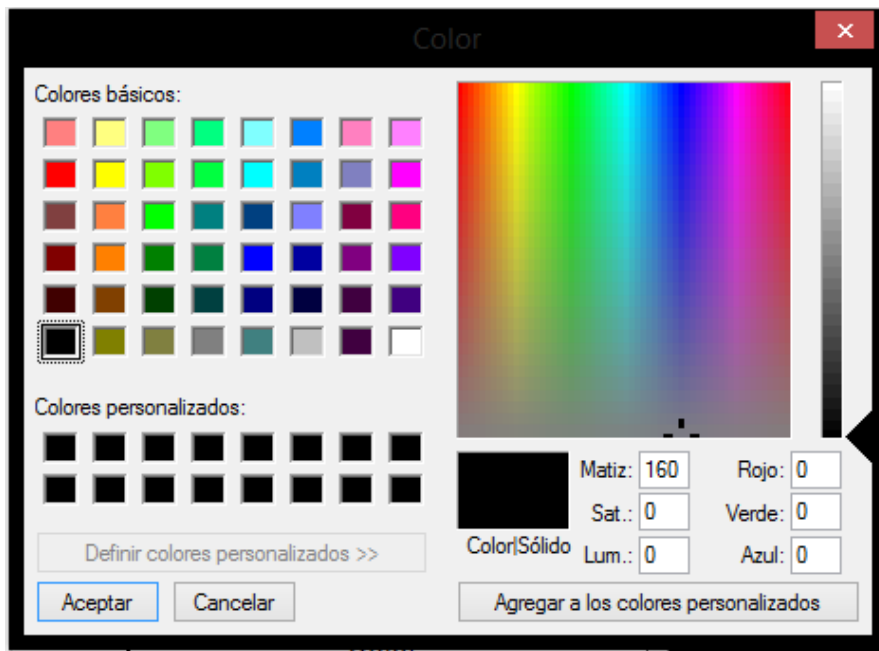


Figura 45. Selección de color.

5.3.2. Relación

Al escoger la opción de “Relación” se abre una ventana emergente con los campos Grosor de línea y Color de línea como se observa en la Figura 46.

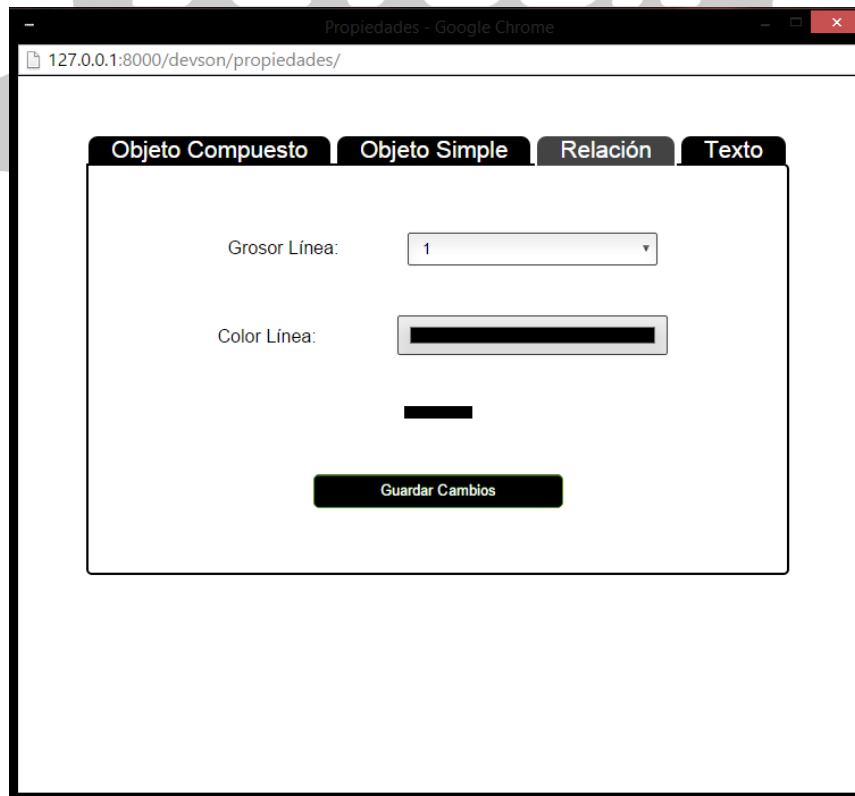


Figura 46. Cualidades relación.

En el campo Grosor de línea se despliega una lista con tamaños de grosor que van desde 1 hasta 10 píxeles (ver Figura 47), en el campo de Color de línea se presiona el botón que se encuentra al lado de la etiqueta y aparecerá un selector de color para elegir de una gran variedad de tonos (ver Figura 45); los cambios se visualizan en el objeto de ejemplo dentro de la ventana.



Figura 47. Selección de grosor.

5.3.3. Texto

Al escoger la opción de “Texto” se abre una ventana emergente con los campos Fuente, Tamaño y Color como se observa en la Figura 48.

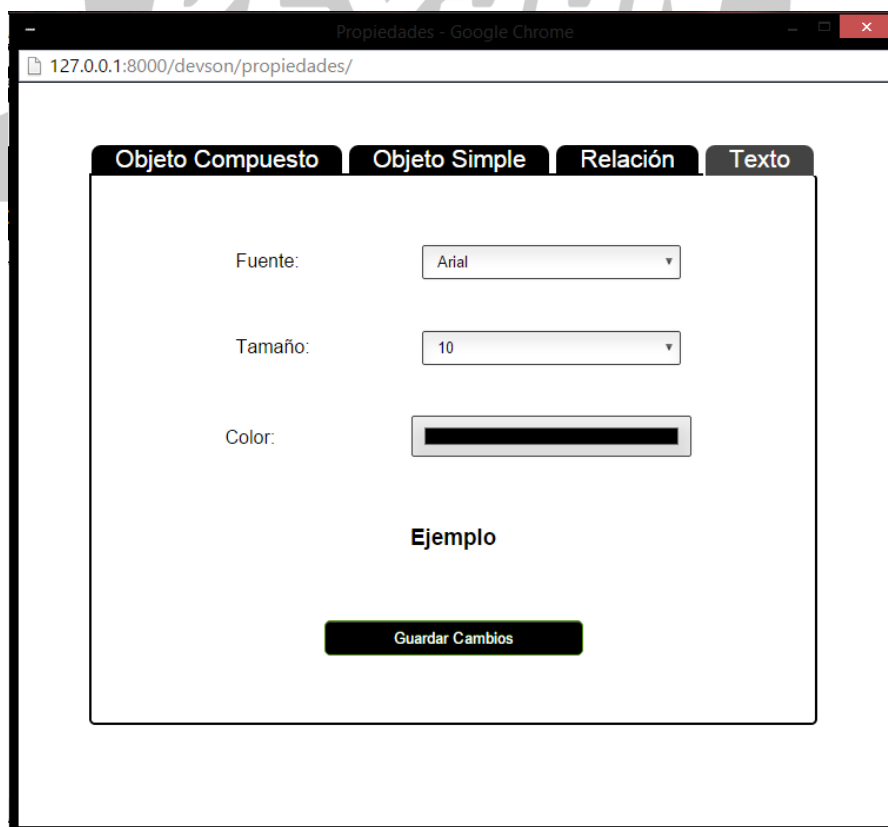


Figura 48. Cualidades texto

En el campo Fuente se despliega una lista con los tipos de letra más utilizados (ver Figura 49), en el campo Tamaño se despliega una lista con tamaños que van desde 10 hasta 20 (ver Figura 50), en el campo de Color se presiona el botón que se encuentra al lado de la etiqueta y aparecerá un selector de color para elegir de una gran variedad de tonos (ver Figura 45); los cambios se visualizan en el objeto de ejemplo dentro de la ventana.

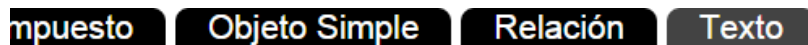


Figura 49. Selección de fuente.

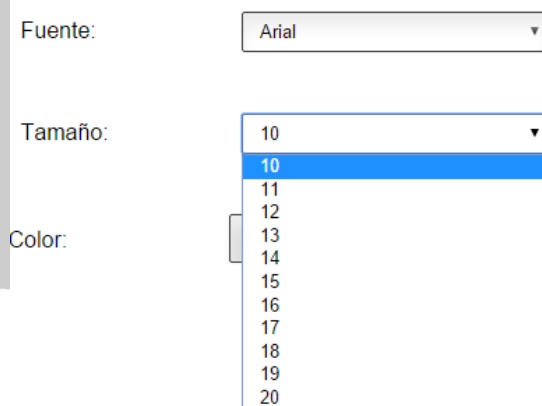
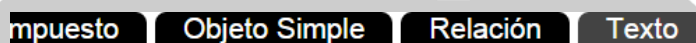


Figura 50. Selección tamaño.

5.3.4. Ejemplo

En un caso de ejemplo, si se desea ajustar los objetos simples con un grosor de 2 píxeles, un color de línea naranja y un color de relleno verde; los objetos compuestos con un grosor de 2 píxeles, un color de línea azul y un color de relleno fucsia; las relaciones con un grosor de 2 píxeles y color gris y el texto con fuente Arial 17 de color verde oscuro (Ver Figura 51, Figura 52, Figura 53 y Figura 54) se obtiene un diagrama similar al mostrado en la Figura 55.

Objeto Compuesto**Objeto Simple****Relación****Texto**

Grosor Línea:

2

Color Línea:

Color:

Guardar Cambios

Figura 51. Ajustes objetos simples.

Objeto Compuesto**Objeto Simple****Relación****Texto**

Grosor Línea:

2

Color Línea:

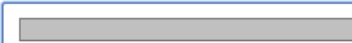
Color:

Guardar Cambios

Figura 52. Ajustes objetos compuestos.

Objeto Compuesto**Objeto Simple****Relación****Texto**

Grosor Línea: 2

Color Línea: 



Guardar Cambios

Figura 53. Ajustes relaciones.

Objeto Compuesto**Objeto Simple****Relación****Texto**

Fuente: Arial

Tamaño: 17

Color: 

Ejemplo

Guardar Cambios

Figura 54. Ajustes de texto.

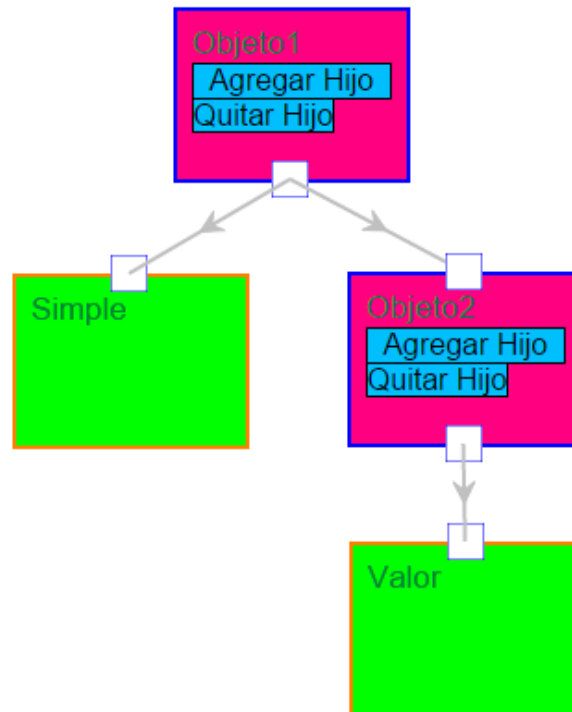



Figura 55. Resultado de ajustes.

5.4. Guardar un proyecto

Para guardar un proyecto, se accede desde el menú Archivo → opción Guardar ó desde el botón “Guardar proyecto” . Se abrirá una ventana emergente en la cual se encuentran las opciones de guardar en la base de datos o guardar en el equipo en donde se esté trabando (ver Figura 56).

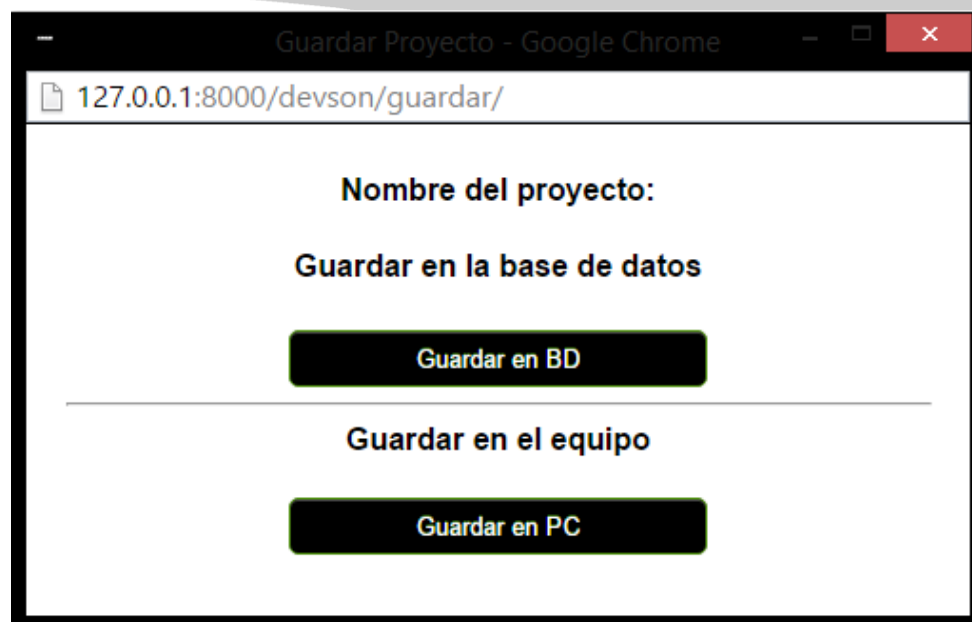


Figura 56. Ventana de guardar.

Si se escoge la opción de guardar en la base de datos, se presiona el botón “Guardar en BD” y el contenido de la ventana cambiará como se muestra en la imagen 57 indicando el éxito del proceso.

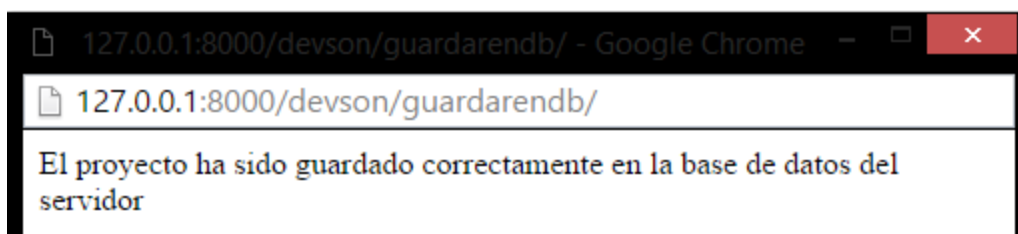


Figura 57. Proyecto guardado en BD.

Si se escoge la opción de guardar en el equipo, se presiona el botón “Guardar en PC” y el navegador descargará un archivo cuyo contenido puede ser interpretado por el aplicativo en caso que se requiera abrir desde otro ordenador (ver Figura 58).

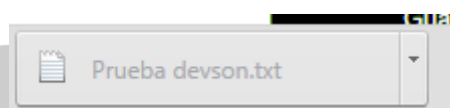



Figura 58. Proyecto descargado desde chrome.

5.5. Abrir un proyecto

Para guardar un proyecto, se accede desde el menú Archivo → opción Abrir ó desde el botón “Abrir proyecto” . Se abrirá una ventana emergente en la cual se encuentran las opciones de abrir proyectos almacenados en la base de datos o abrir los proyectos almacenados en el equipo en forma de archivos.

Si se escoge la opción de abrir proyectos en la base de datos, se elige de la lista de proyectos almacenados por el usuario y se presiona el botón “Abrir proyecto del servidor” (ver Figura 59).

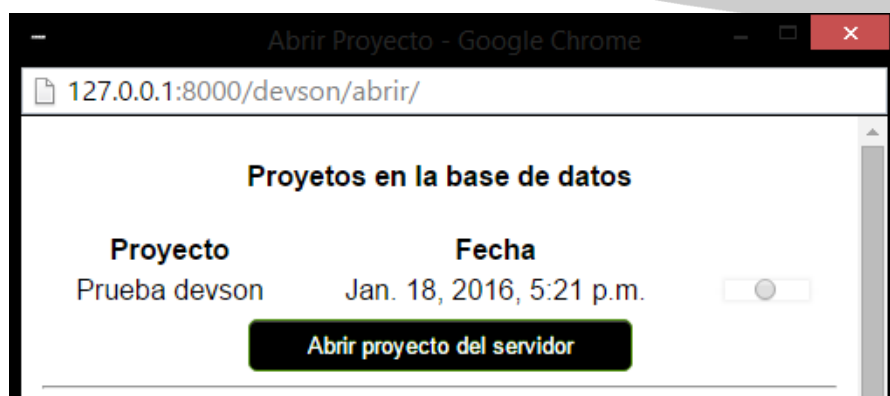


Figura 59. Abrir proyecto del servidor.

Si se escoge la opción de abrir proyectos en el equipo, se presiona el botón “Examinar” (ver Figura 60), se escoge el archivo que contiene el proyecto (ver Figura 61) y se presiona el botón “Abrir proyecto local” (ver Figura 62).

Proyectos en el equipo

Examinar

Abrir proyecto local

Figura 60. Formulario para abrir proyectos desde el equipo.

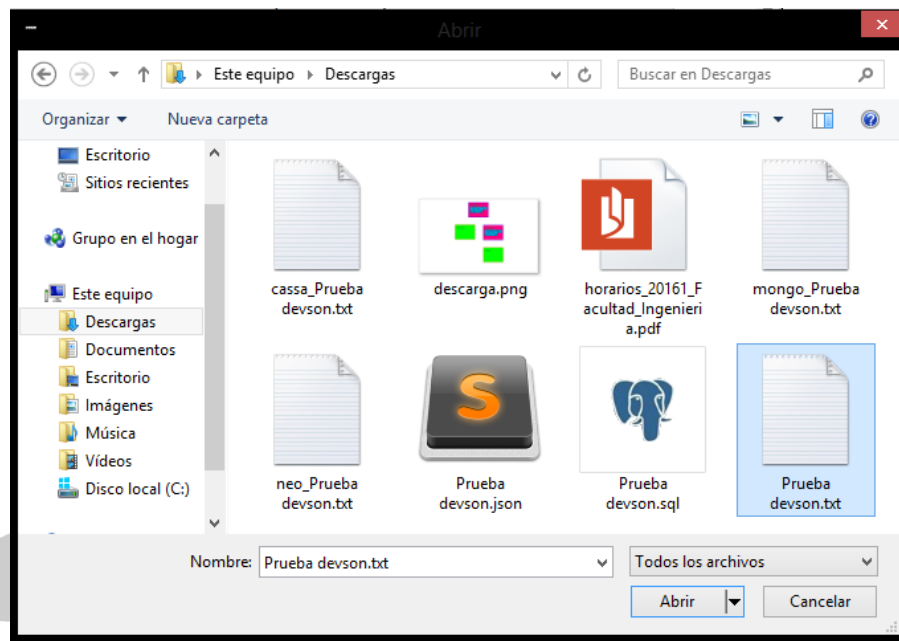


Figura 61. Elección del archivo.

Proyectos en el equipo

C:\fakepath\Prueba devson.txt

Examinar

Abrir proyecto local

Figura 62. Archivo ubicado.

El contenido de la página principal cambiará acorde al contenido del proyecto tanto en el área de trabajo como en el panel de proyecto se muestra en la imagen 63 indicando el éxito del proceso.

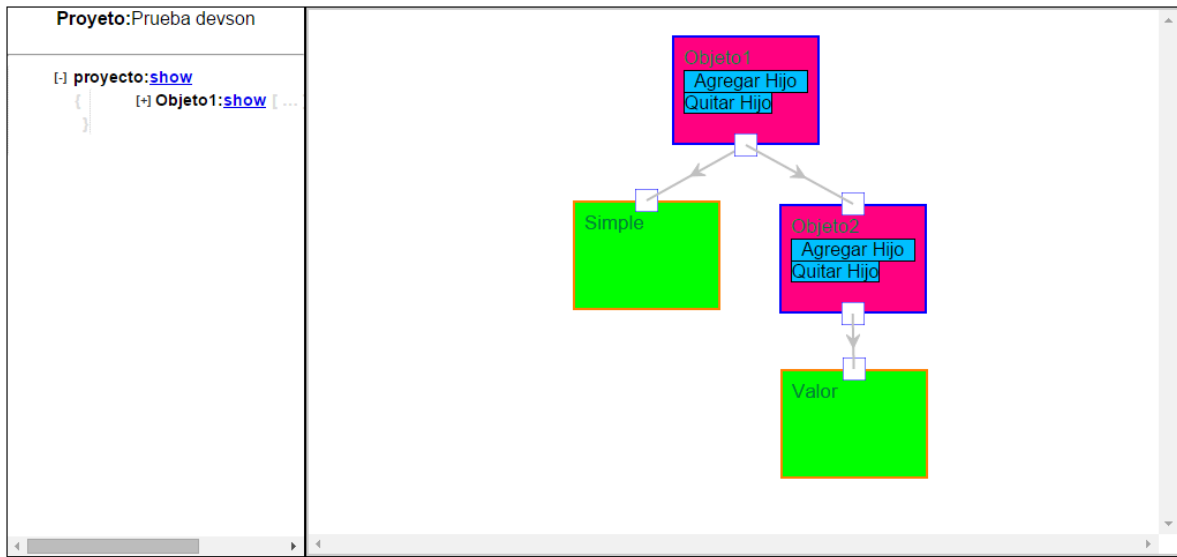



Figura 63. Proyecto abierto.

5.6. Exportar

Al momento de exportar el proyecto, se presentan varios formatos dependiendo del uso por el que se ha creado el proyecto; entre estos se encuentran:

- Imagen PNG.
- JSON.
- SQL.
- Cassandra.
- MongoDB.
- Neo4J.

5.6.1. Exportar como imagen

Para guardar el proyecto como una imagen, se accede desde el botón “Imprimir como imagen” . Se descargará un archivo con formato PNG desde el navegador (ver Figura 64 y Figura 65).

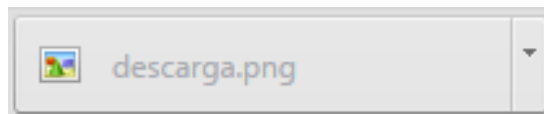


Figura 64. Archivo descargado desde chrome.

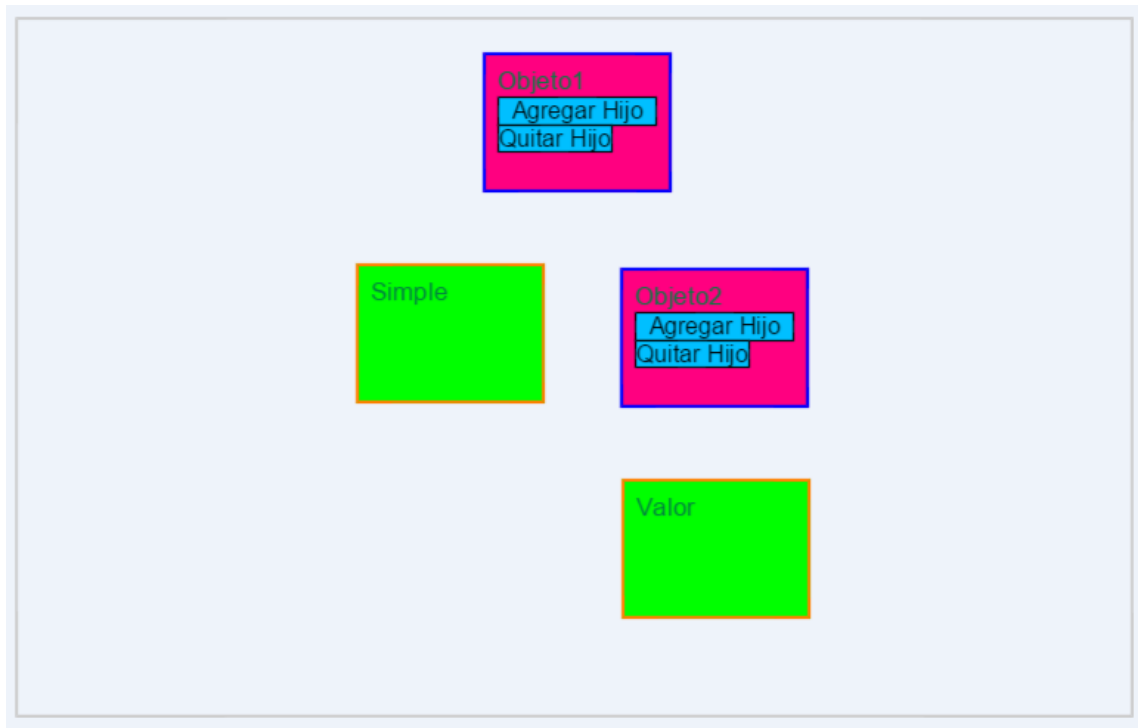



Figura 65. Imagen obtenida.

5.6.2. Exportar como JSON

Para guardar el proyecto como una estructura de datos en JSON (por ejemplo el visualizado en la Figura 66), se accede desde el menú Archivo → opción Exportar como → opción JSON ó desde el botón “Exportar como JSON” . Se descargará un archivo con formato JSON desde el navegador (ver Figura 67 y Figura 68).

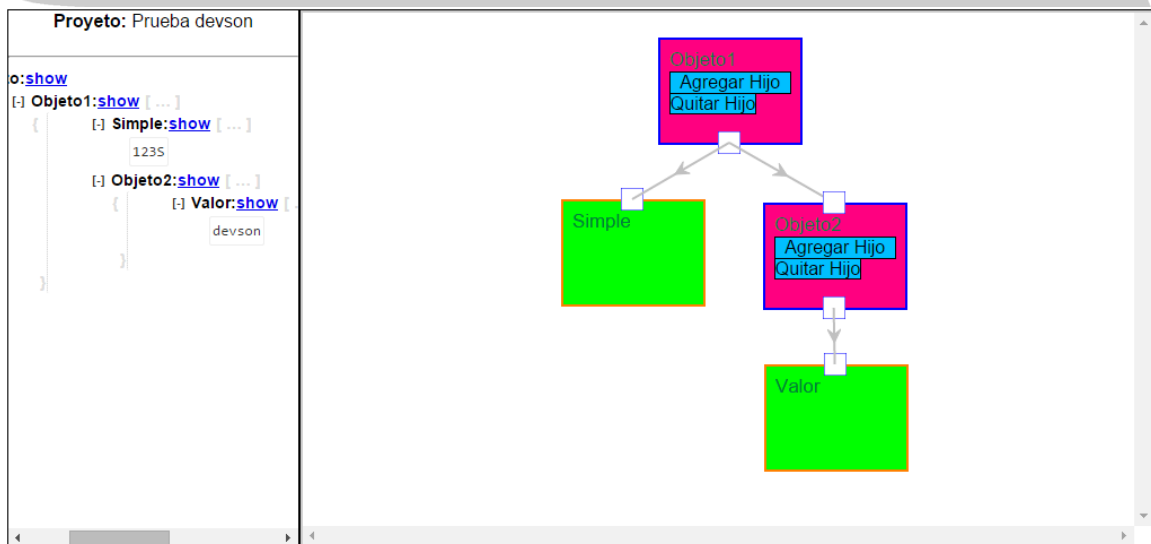


Figura 66. Ejemplo de estructura de datos.

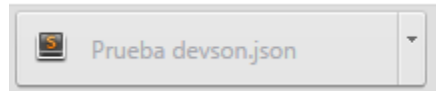


Figura 67. Archivo descargado desde chrome.

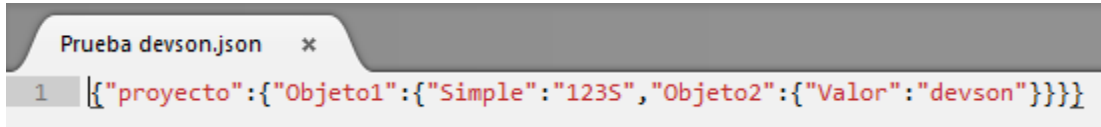


Figura 68. Archivo JSON.

5.6.3. Exportar como SQL

Para guardar el proyecto como una estructura de datos en SQL (por ejemplo el visualizado en la Figura 66), se accede desde el menú Archivo → opción Exportar como → opción SQL. Se descargará un archivo con formato SQL desde el navegador en donde se establecen en forma de tablas la estructura creada (ver Figura 69 y Figura 70).

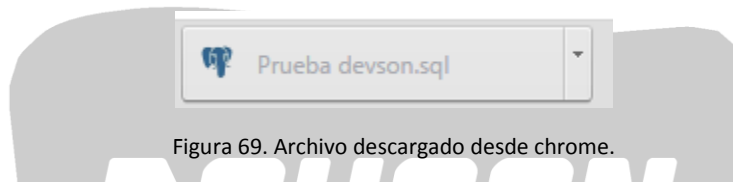


Figura 69. Archivo descargado desde chrome.

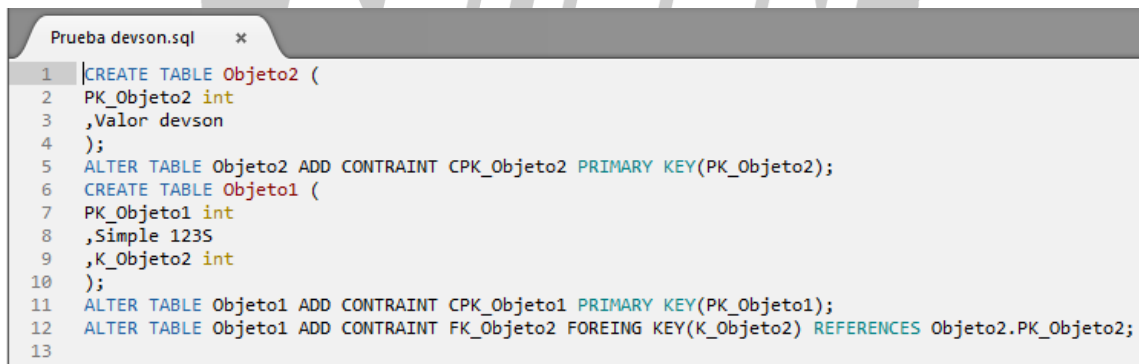


Figura 70. Archivo SQL.

5.6.4. Exportar como Cassandra

Para guardar el proyecto como una estructura de datos en Cassandra (por ejemplo el visualizado en la Figura 66), se accede desde el menú Archivo → opción Exportar como → opción NoSQL → opción Cassandra. Se descargará un archivo con formato TXT desde el navegador en donde se establecen en forma de tablas la estructura creada (ver Figura 71 y Figura 72).

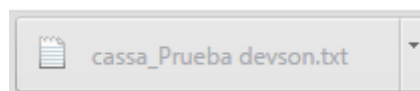
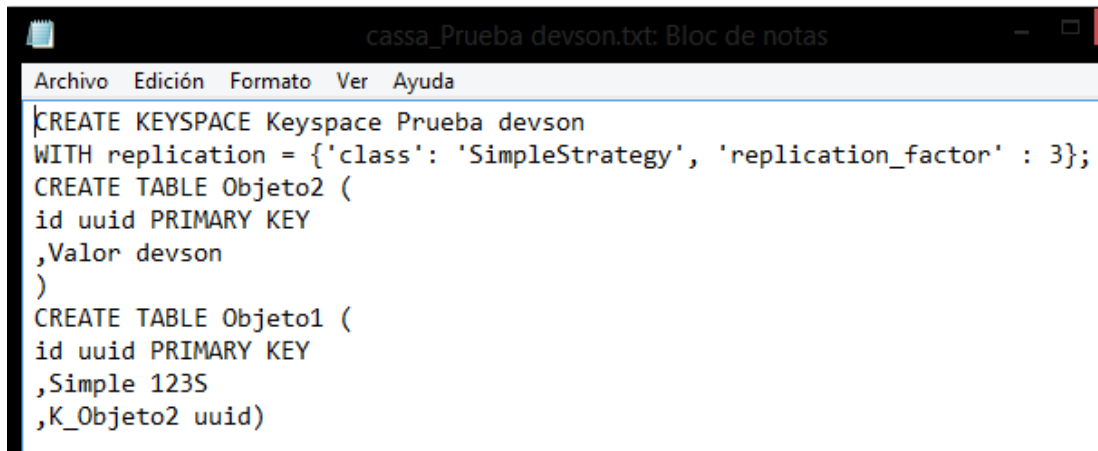


Figura 71. Archivo descargado desde chrome.



```
CREATE KEYSPACE Keyspace Prueba devson
WITH replication = {'class': 'SimpleStrategy', 'replication_factor' : 3};
CREATE TABLE Objeto2 (
id uuid PRIMARY KEY
,Valor devson
)
CREATE TABLE Objeto1 (
id uuid PRIMARY KEY
,Simple 123S
,K_Objeto2 uuid)
```

Figura 72. Archivo Cassandra.

5.6.5. Exportar como MongoDB

Para guardar el proyecto como una estructura de datos en MongoDB (por ejemplo el visualizado en la Figura 66), se accede desde el menú Archivo → opción Exportar como → opción NoSQL → opción MongoDB. Se descargará un archivo con formato TXT desde el navegador en donde se establece en forma de colección la estructura creada (ver Figura 73 y Figura 74).

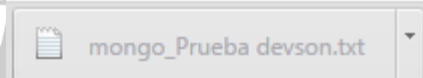
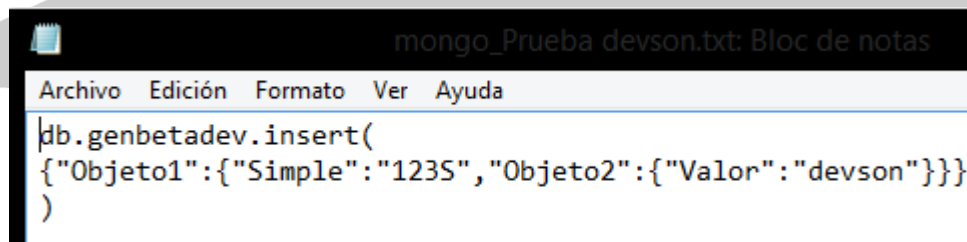


Figura 73. Archivo descargado desde chrome.



```
db.genbetadev.insert(
{'Objeto1':{'Simple':'123S'}, 'Objeto2':{'Valor':'devson'}}
)
```

Figura 74. Archivo MongoDB.

5.6.6. Exportar como Neo4J

Para guardar el proyecto como una estructura de datos en Neo4J (por ejemplo el visualizado en la Figura 66), se accede desde el menú Archivo → opción Exportar como → opción NoSQL → opción Neo4J. Se descargará un archivo con formato TXT desde el navegador en donde se establecen en forma de nodos la estructura creada (ver Figura 75 y Figura 76).

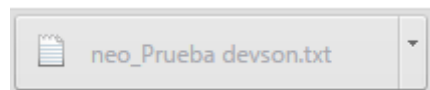


Figura 75. Archivo descargado desde chrome.

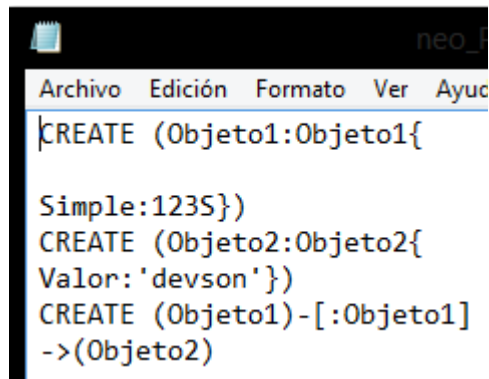



Figura 76. Archivo Neo4J.

5.7. Operaciones complementarias


Como operaciones complementarias se encuentran los procesos básicos de una herramienta, entre estos se encuentran:

- Copiar objeto.
- Pegar objeto.
- Eliminar objeto.
- Deshacer.
- Rehacer.

5.7.1. Copiar objeto

Para copiar un objeto, se debe hacer clic sobre el objeto deseado y luego se accede a la opción copiar desde el menú Edición → opción Copiar ó desde el botón “Copiar objeto” .

5.7.2. Pegar objeto

Para pegar un objeto, se debe hacer previamente el proceso de copiar objeto y luego se accede a la opción pegar desde el menú Edición → opción Pegar ó desde el botón “Pegar objeto” .

Por ejemplo, si se desea duplicar un objeto compuesto se hace clic sobre él, se copia y se pega; al hacerlo un objeto con las mismas características aparecerá en la esquina superior izquierda del área de trabajo y los datos se duplicarán dentro del panel de proyecto (ver Figura 77 y Figura 78).

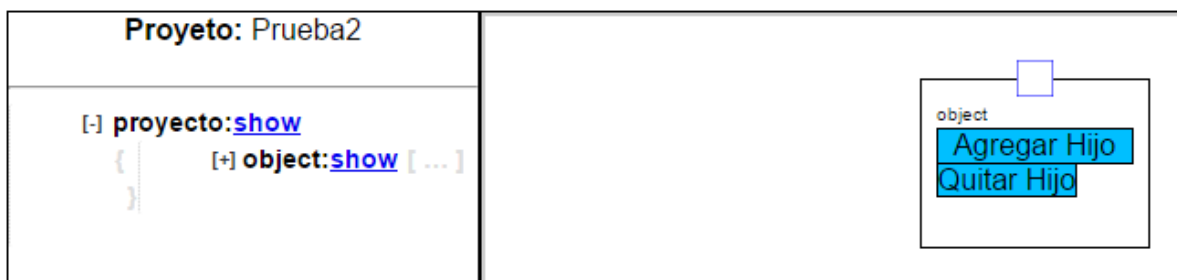


Figura 77. Objeto compuesto a copiar.

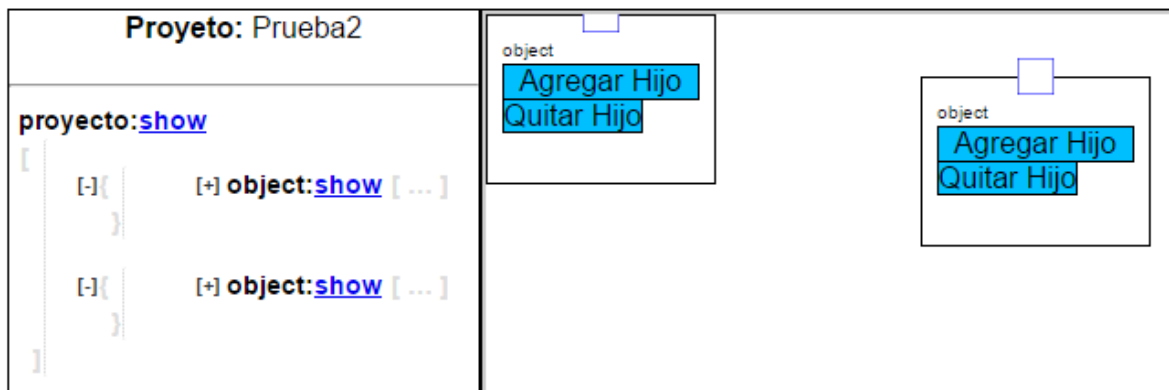


Figura 78. Objeto compuesto pegado.

Por ejemplo, si se desea duplicar un objeto simple se hace clic sobre él, se copia y se pega; al hacerlo un objeto con las mismas características aparecerá en la esquina superior izquierda del área de trabajo y los datos se duplicarán dentro del panel de proyecto (ver Figura 79 y Figura 80).

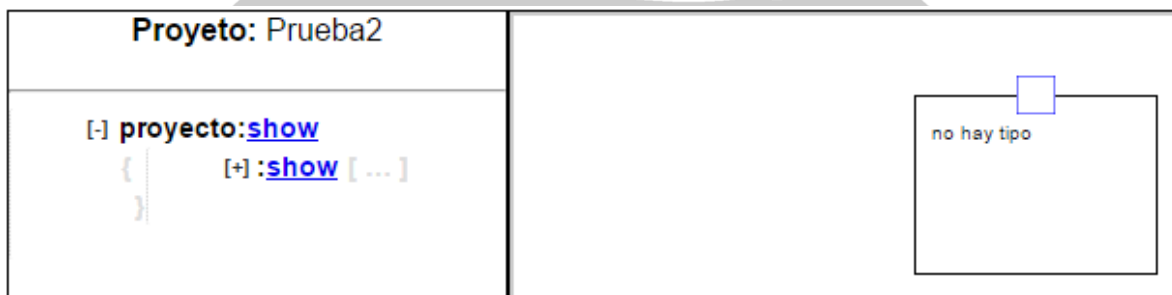



Figura 79. Objeto simple a copiar.



Figura 80. Objeto simple pegado.

5.7.3. Eliminar objeto

Para eliminar un objeto, se debe hacer previamente clic sobre el elemento deseado y luego se accede a la opción de eliminar desde el botón “Eliminar objeto” .

Por ejemplo, si se desea duplicar el objeto B del diagrama de la Figura 81, se hace clic sobre él y luego se presiona el botón “Eliminar objeto”; al hacerlo el objeto desaparece del área de trabajo y sus datos se eliminan del panel de proyecto actualizando su estructura sin afectar los valores de los demás nodos (ver Figura 82).

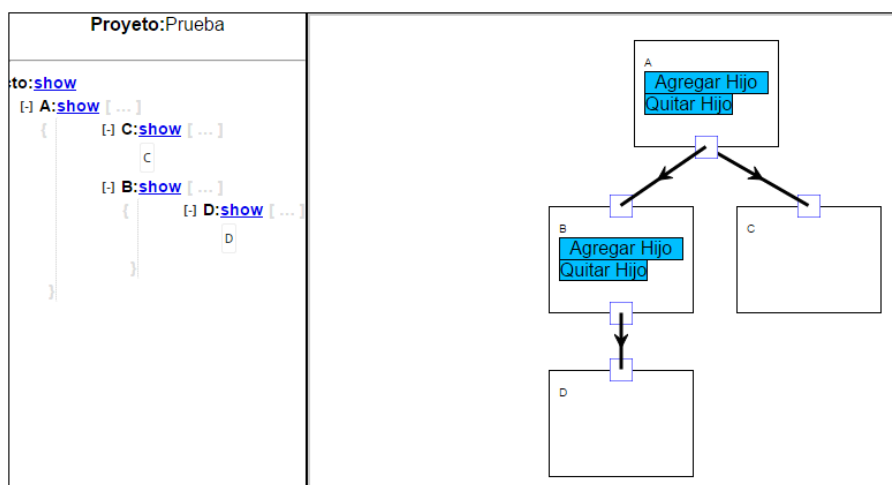


Figura 81. Ejemplo a eliminar.

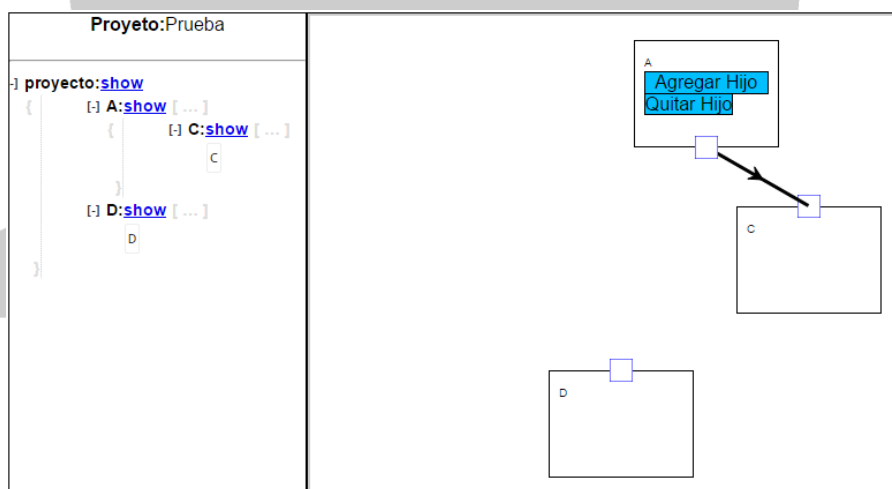




Figura 82. Elemento eliminado.

5.7.4. Deshacer

Para revertir alguna acción se debe presionar el botón “Deshacer” , se visualiza que el área de trabajo y el panel de proyecto cambian acorde a los cambios hechos.

5.7.5. Rehacer

Para revertir alguna acción se debe presionar el botón “Rehacer” , se visualiza que el área de trabajo y el panel de proyecto cambian acorde a los cambios hechos.

REFERENCIAS Y BIBLIOGRAFÍA

- Acenswhitepapers, C. d. (2 de 2014). *acens.com*. (Una compañía de Telefonica)
Recuperado el 2 de 4 de 2015, de acens.com: <http://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf>
- Artavia, L. R., & Villalobos, M. (s.f.). *Universidad Latinoamericana de Ciencia y Tecnología*.
Recuperado el 10 de Junio de 2015, de Relación entre las bases de datos NoSQL y Big Data: <http://bb9.ulacit.ac.cr/tesinas/Publicaciones/043241.pdf>
- Colaboradores de MongoDB. (s.f.). *JSON AND BSON*. Recuperado el 29 de Octubre de 2014, de <http://www.mongodb.com/json-and-bson>
- ECMA International. (s.f.). *THE JSON DATA INTERCHANGE FORMAT*. Recuperado el 29 de Octubre de 2014, de <http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf>
- Escofet, C. M. (s.f.). *El Lenaguje SQL*. España: Universidad Abierta de Catalunya.
- Gracia del Busto, H., & Yanes Enríquez, O. (2012). *Revista Telemática*. Recuperado el 10 de Junio de 2015, de Revista Telemática: <http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/74/74>
- JSON. (s.f.). *JSON*. Recuperado el 17 de Octubre de 2014, de <http://www.json.org/json-es.html>
- Robie, Fourny, G., Brantner, M., Florescu, D., Westmann, T., & Zaharioudakis, M. (s.f.). *jsoniq.org*. Recuperado el 8 de 3 de 2015, de jsoniq.org: <http://www.jsoniq.org/>
- Robie, J., Ghislain, F., Matthias, B., Florescu, D., Westmann, T., & Zaharioudakis, M. (22 de 7 de 2013). *JSONiq Use Cases*. Recuperado el 8 de 3 de 2015, de JSONiq Use Cases: <http://www.jsoniq.org/docs/JSONiq-usecases/html-single/>