

16

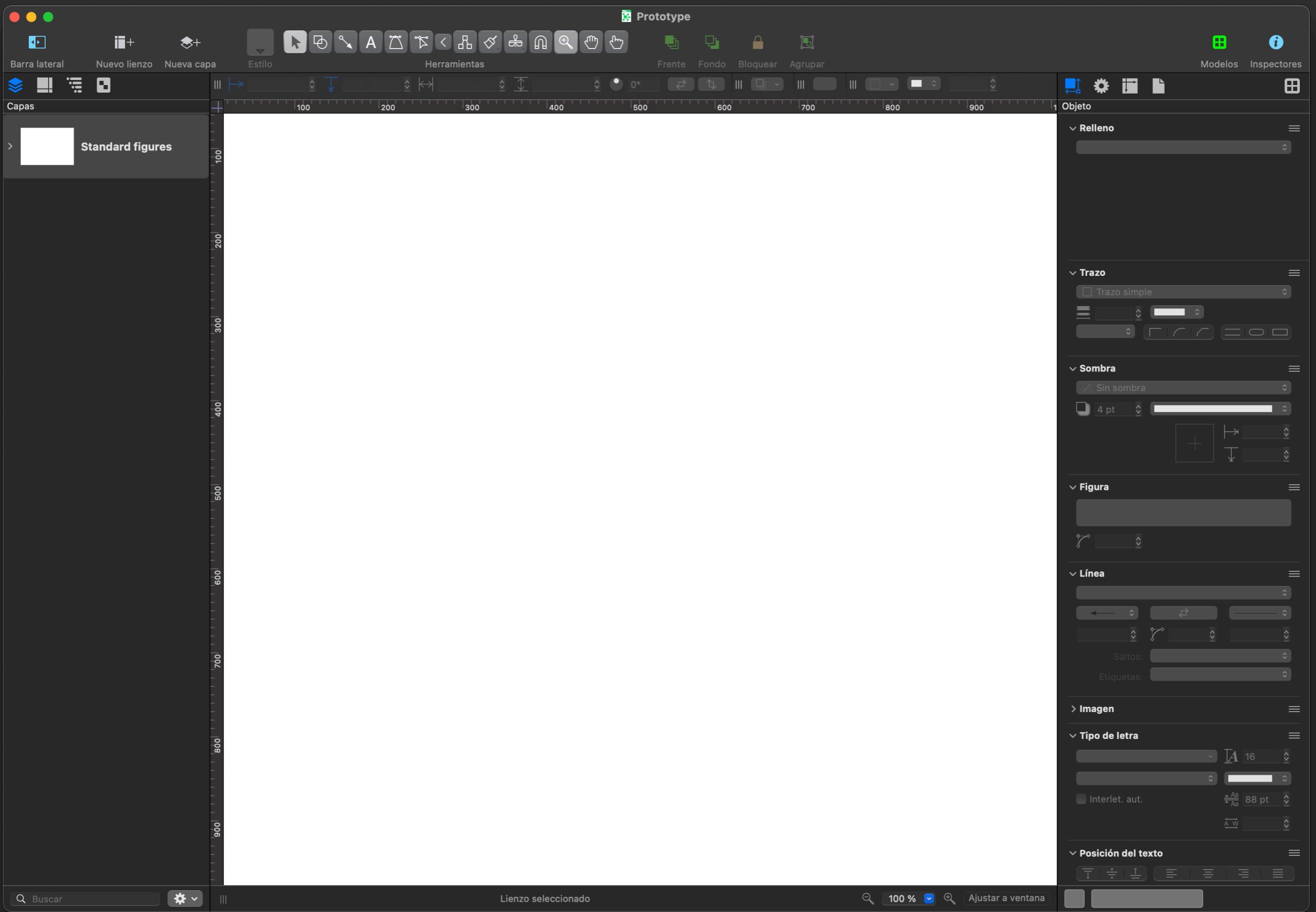
Prototype

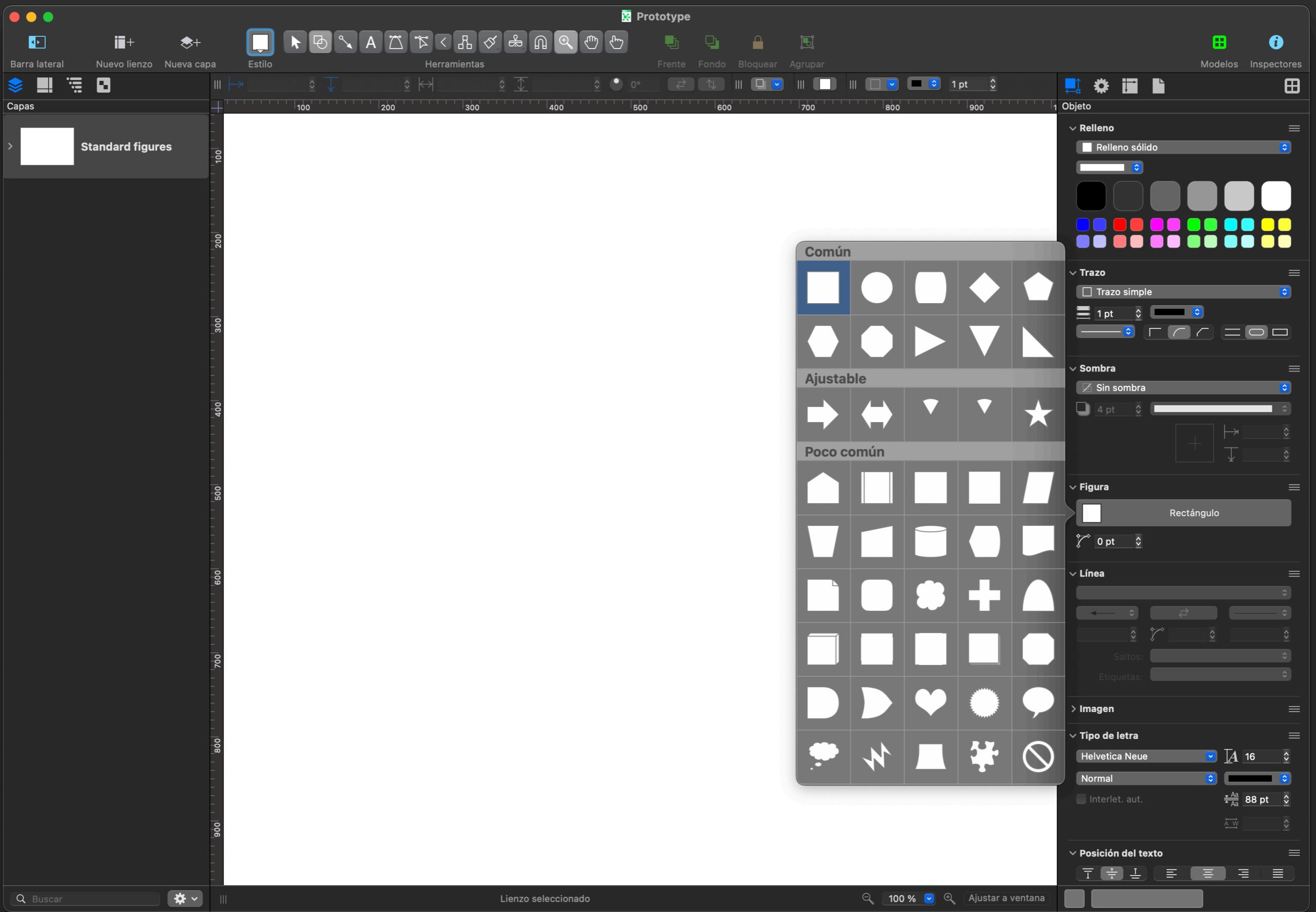
Software Design

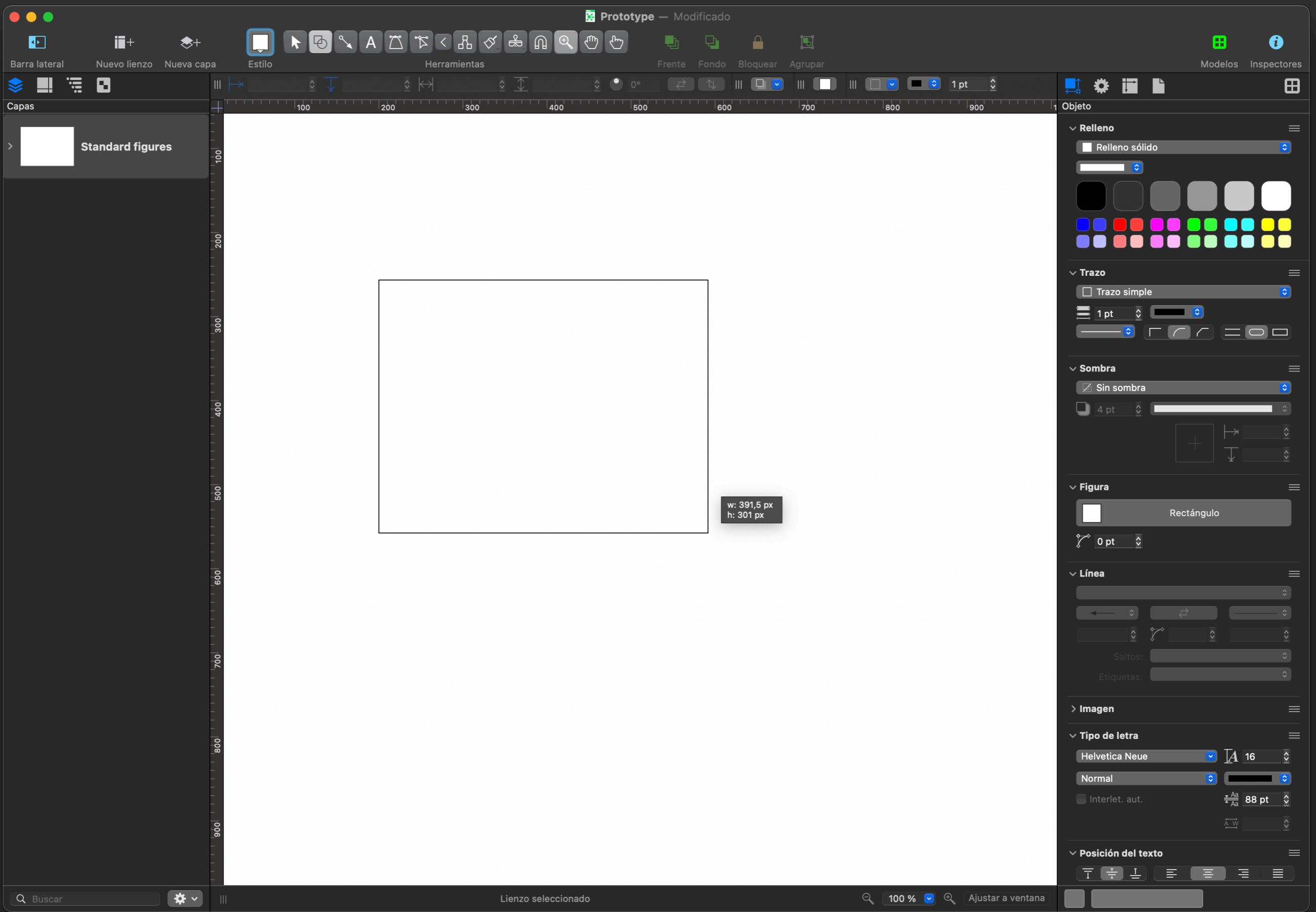
Bachelor's Degree in Computer Science - Software Engineering

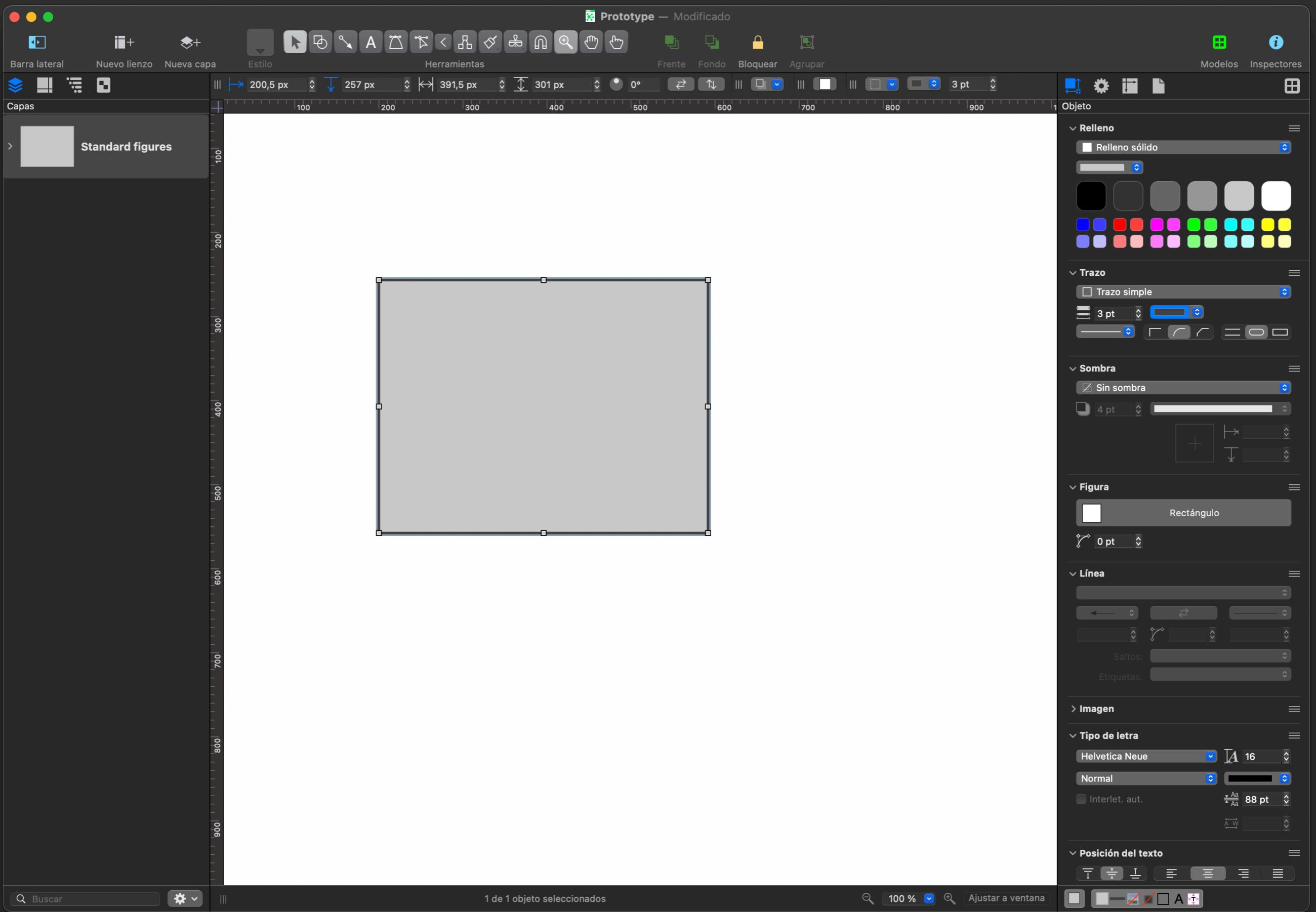
2024-2025

Veamos un posible ejemplo de aplicación del patrón tomando como ejemplo una herramienta de creación de diagramas (concretamente, OmniGraffle, para Mac).









¿Qué hemos hecho?

Hasta aquí, nada que no supiéramos cómo diseñar.

Hemos creado uno de los tipos de figuras predefinidos (en este caso, un rectángulo).

Muy probablemente, cada uno de ellos se modelaría como clases distintas.

No es lo mismo dibujar un rectángulo que un círculo, un cubo, una línea o cualquier otro símbolo predefinido.

Figuras definidas por el usuario

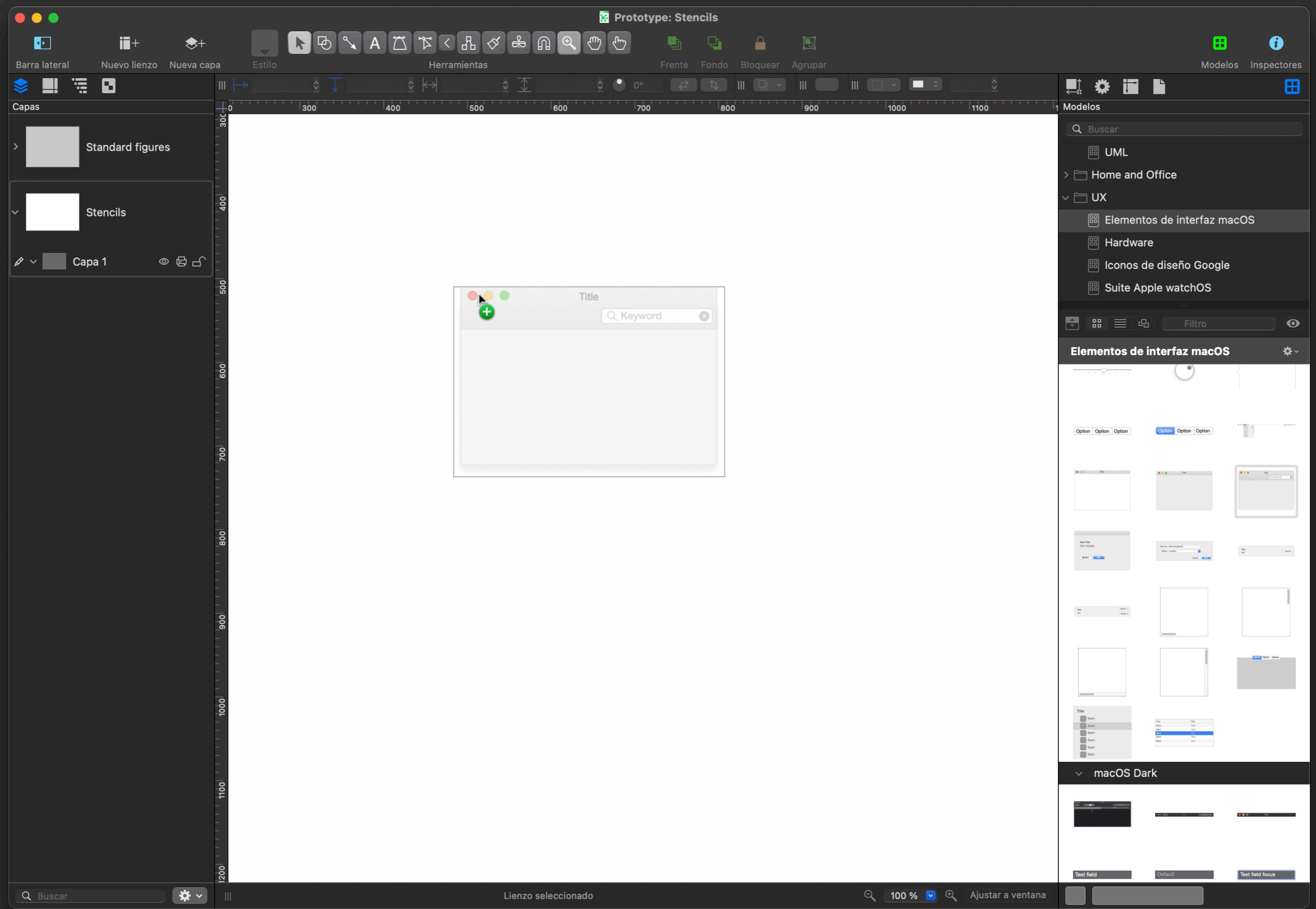
Pero resulta que el programa debe permitir crear nuevos tipos de figuras al usuario.

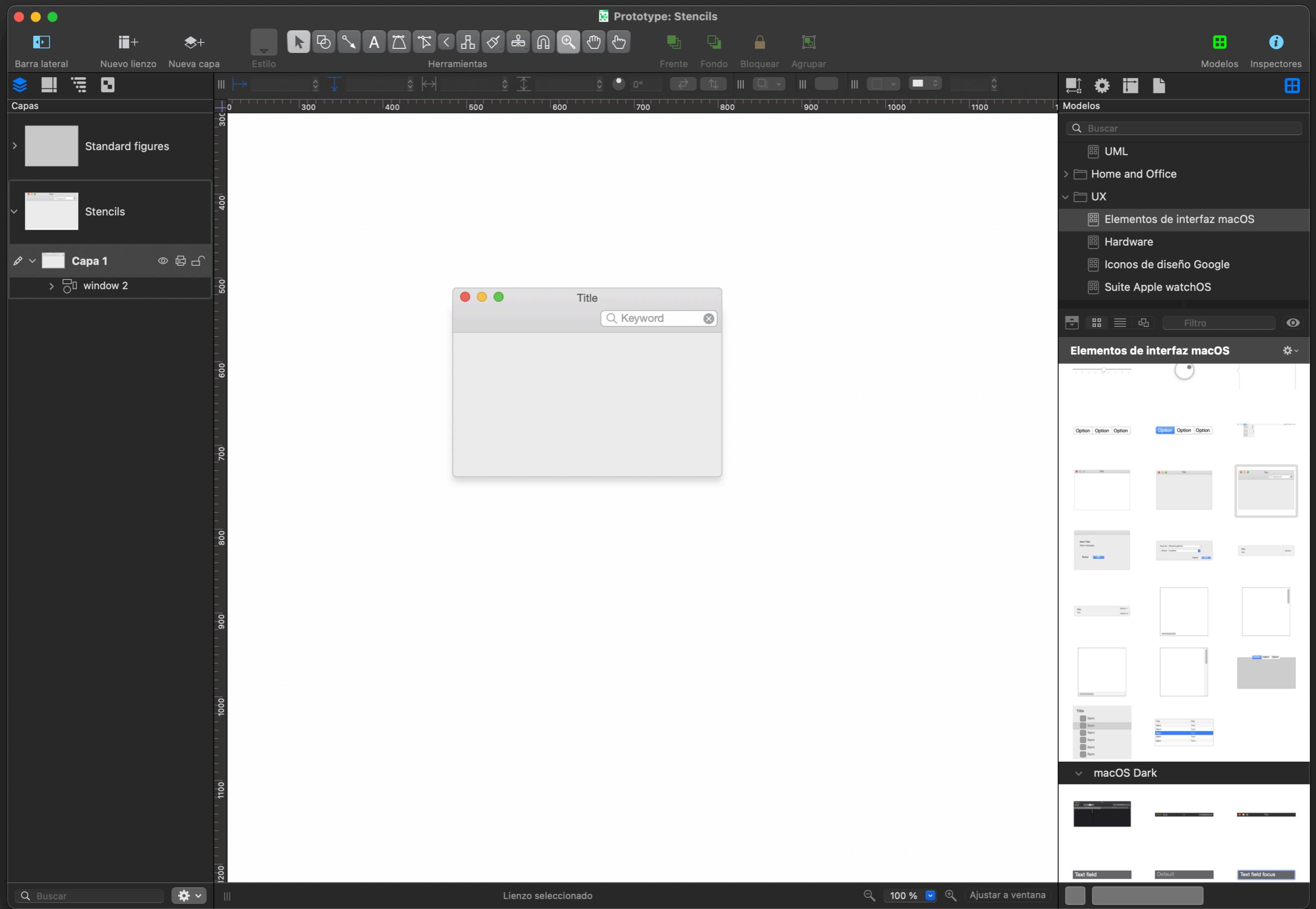
Elementos de diagramas de UML

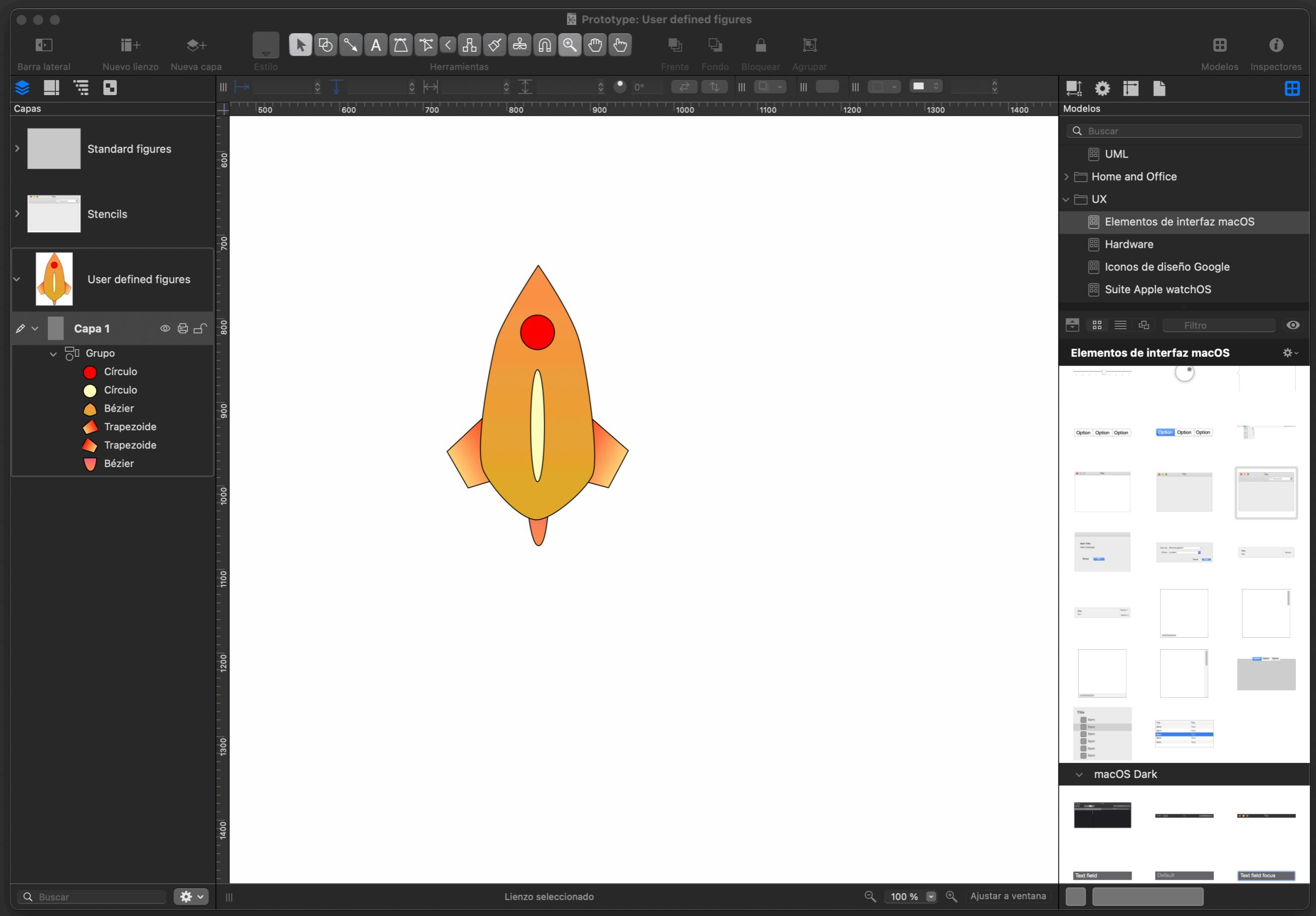
Componentes para elaborar bocetos de interfaces de usuario

Cualquier otra figura compleja que queramos guardar como una plantilla para poder usarla posteriormente

Ha de tenerse en cuenta que esto no puede hacerse programando nuevas clases; tiene que poder hacerlo directamente el usuario final, a través de la interfaz de usuario (es decir, simplemente dibujando).







**El programa, obviamente, no
conoce dichos tipos de figura.**

Se crean en tiempo de ejecución.
¡Son infinitos!

Tenemos que lograr usar esas figuras definidas por el usuario como plantillas.

En primer lugar, habrá que guardarlas.

Como cualquier otro objeto.

Y a continuación debemos poder usarlas para crear nuevas figuras a partir de ellas, tal como hacemos con las predefinidas.

Para ello deben saber cómo clonarse.

Tenemos que lograr usar esas figuras definidas por el usuario como plantillas.

En primer lugar, habrá que guardarlas.

Como cualquier otro objeto.

Y a continuación debemos poder usarlas para crear nuevas figuras a partir de ellas, tal como hacemos con las predefinidas.

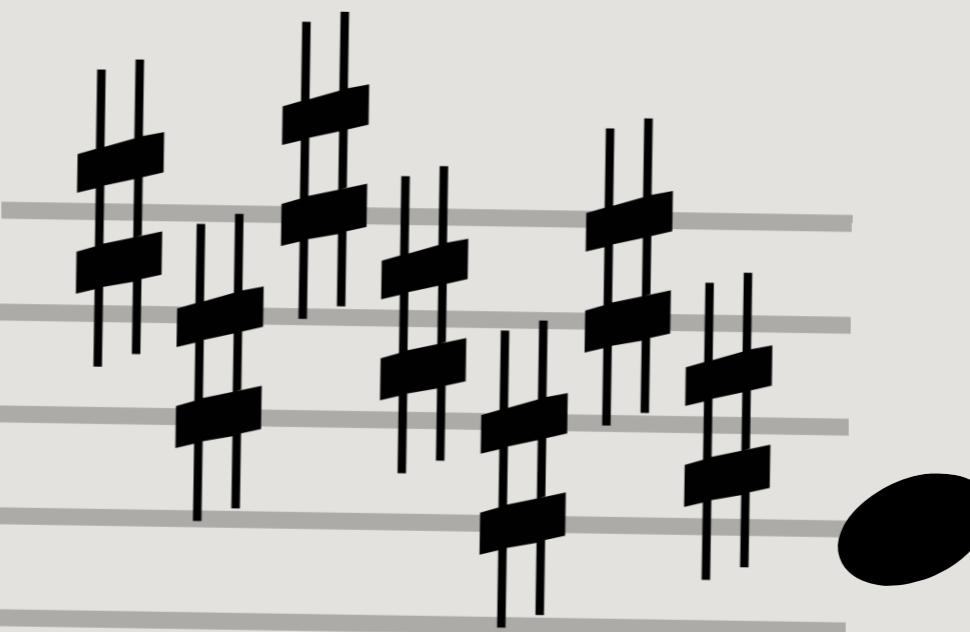
Para ello deben saber cómo clonarse.

Prototype

Patrón de creación, de objetos

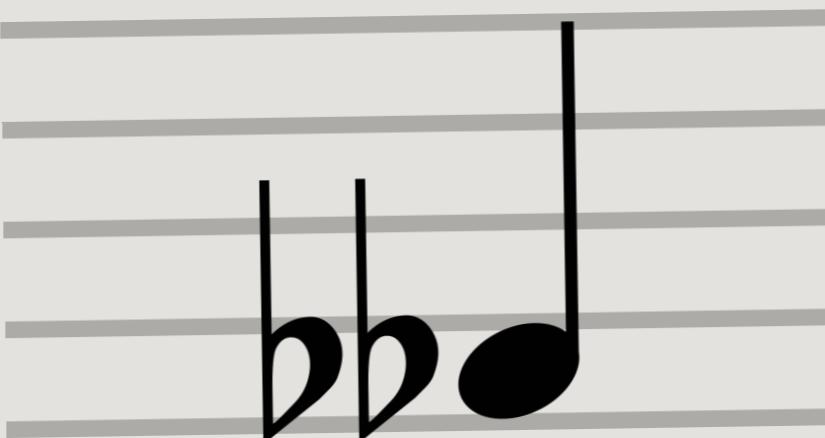
Especifica los tipos de objetos a crear usando una instancia prototípica, y crea nuevos objetos copiando dicho prototipo.

Motivación



= 120

Queremos crear un editor de partituras musicales particularizando un framework general de editores gráficos y añadiéndole nuevos objetos que representen las notas, silencios, claves, ligaduras...



Supongamos que el framework provee una interfaz Graphic para los elementos gráficos.

Además, proporciona una interfaz Tool para crear herramientas como las predefinidas de la paleta.

Por último, el framework también cuenta con una subclase GraphicTool predeterminada para crear instancias de objetos gráficos y añadirlas al documento.

Pero dicha clase `GraphicTool` representa un problema como diseñadores del framework.

Las clases para notas y pentagramas son específicas de nuestra aplicación, pero la clase `GraphicTool` pertenece al framework: no sabe cómo crear instancias de nuestras clases musicales para añadirlas a la partitura.

Podríamos crear tantas subclases de `GraphicTool` como tipos de símbolos musicales, pero acabaríamos teniendo un montón de clases artificiales que solo se diferenciarían en el tipo de figura a crear.

¿Qué patrón sería, por cierto?

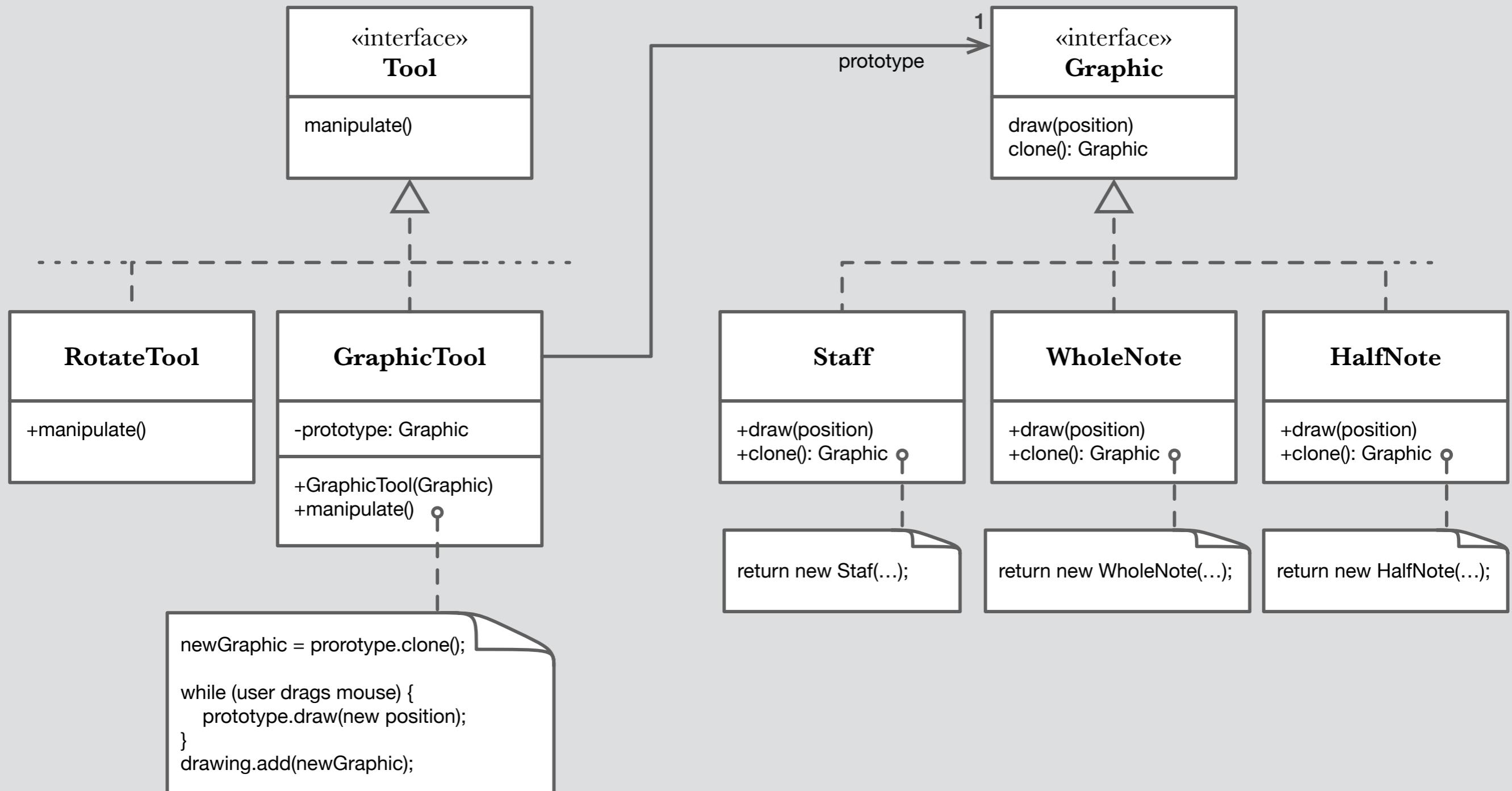
**En vez de eso, ¿no podríamos
parametrizar distintos objetos
GraphicTool con el tipo de figura a crear?**

La solución consiste en hacer que GraphicTool cree un nuevo objeto Graphic copiando («clonando») una instancia de una clase concreta de figura.

Dicha instancia es un prototipo.

Cada objeto GraphicTool recibe en el constructor el prototipo de la figura a clonar y añadir al documento.

Si todas las clases de figuras cuentan con una operación clonar, la herramienta genérica, GraphicTool, podrá crear cualquier tipo de figura.



Aplicabilidad

Úsese el patrón Prototipo cuando un sistema deba ser independiente de cómo se crean, componen y representan sus productos y, además, se de alguna de estas circunstancias.

**Las clases a «instanciar» son
definidas en tiempo de ejecución.**

Como en nuestro ejemplo de las figuras definidas
por el usuario.

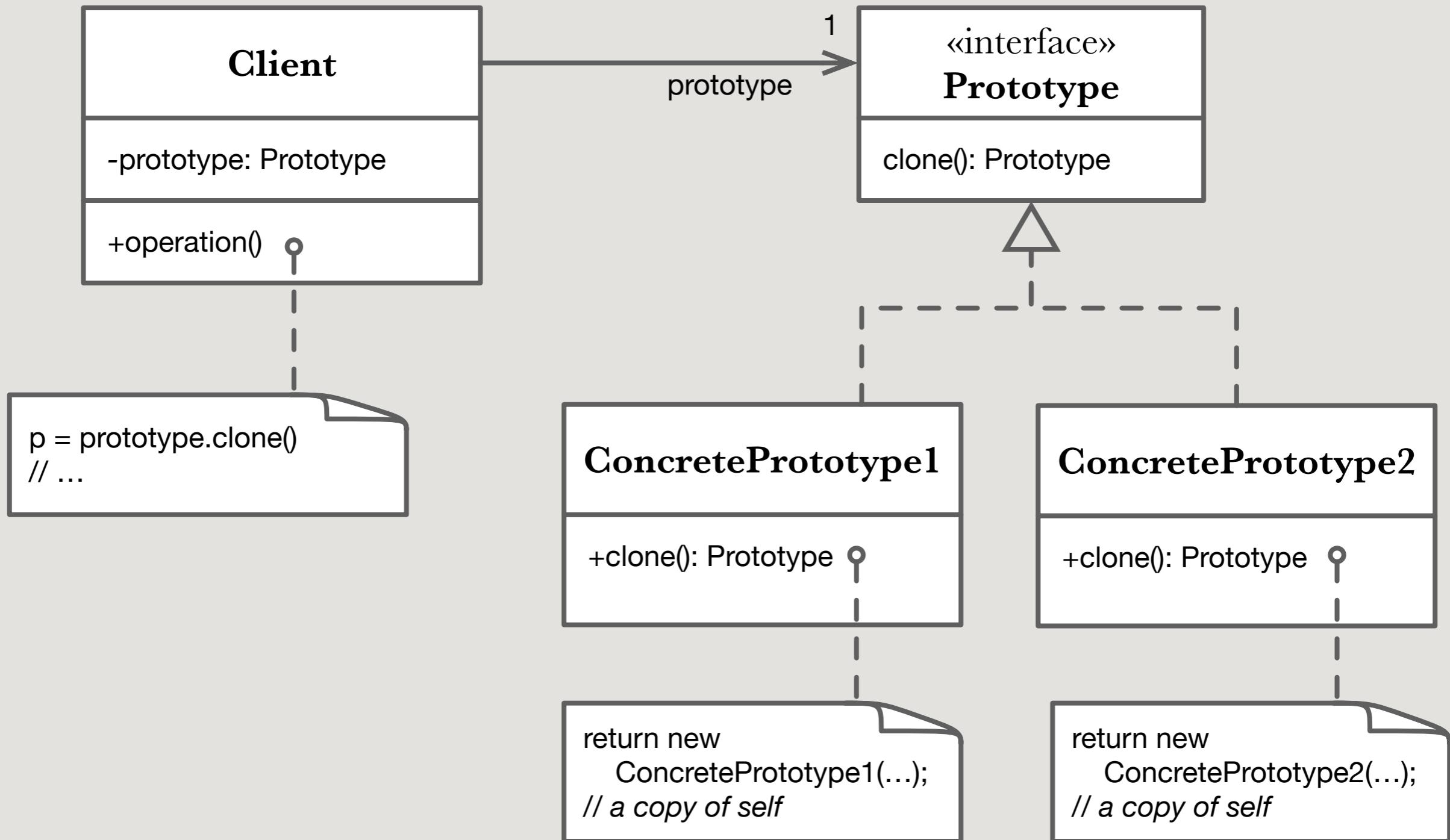
**Para evitar construir una jerarquía
paralela de factorías de productos.**

Cuando esas clases no sean necesarias por otros
motivos.

Cuando las instancias de una clase puedan tener solo unos pocos posibles estados.

Y pueda resultar más conveniente crear los objetos correspondientes como prototipos y clonarlos, en vez de «instanciar» manualmente la clase, cada vez con el estado necesario.

Estructura



La estructura del patrón Prototype

Participantes

Prototype

(Graphic)

Declara una interfaz para clonarse.

ConcretePrototype

(Staff, WholeNote, HalfNote)

Implementa la operación de clonación.

Client

(GraphicTool)

Crea un nuevo objeto pidiéndole al prototipo que se clone.

Colaboraciones

El cliente el pide al prototipo que se clone.

Consecuencias

El Prototype comparte algunas de las consecuencias del Abstract Factory y, en el caso de la variante de jerarquías de clases paralelas (cuando el método de creación es llamado directamente por el cliente), también con el Factory Method:

Oculta las clases concretas de producto al cliente.

Reduciendo así el número de clases con que los clientes deben tratar.

Además, permite que un cliente trabaje con clases desconocidas, concretas de cada aplicación, sin necesidad de hacer cambios.

Otras ventajas adicionales, específicas del patrón son:

Añadir y eliminar productos en tiempo de ejecución

Simplemente registrando el objeto que hará de prototipo en el cliente.

Esto es más flexible que otros patrones de creación, porque un cliente puede instalar y eliminar prototipos en tiempo de ejecución.

Especificar nuevos objetos
modificando valores de sus
propiedades

Mediante composición de objetos, en vez de
creando nuevas clases.

Podemos definir nuevos tipos de objetos «instanciando» clases
existentes y registrando las instancias como prototipos de objetos
cliente.

Autoguardado ⚡ 🏠 🔍 ... Presentación1 Sin etiqueta Buscar (Cmd + Ctrl + U)

Inicio Insertar Dibujar Diseño Transiciones Animaciones Presentación con diapositivas Grabar Revisar Vista Acrobat Grabar Comentarios Presentar en Teams Compartir

Pegar Nueva diapositiva Aptos Display (Títulos) 44 A A A A N K S ab x x AV Aa Imagen Organizar Estilos rápidos Sensibilidad Complementos Diseñador Crear PDF y compartir vínculo

1

Haga clic para agregar título

Haga clic para agregar subtítulo

Haga clic para agregar notas

Diseñador

Haga clic para agregar título

Haga clic para agregar subtítulo

Haga clic para agregar título

Haga clic para agregar subtítulo

Haga clic para agregar título

Haga clic para agregar subtítulo

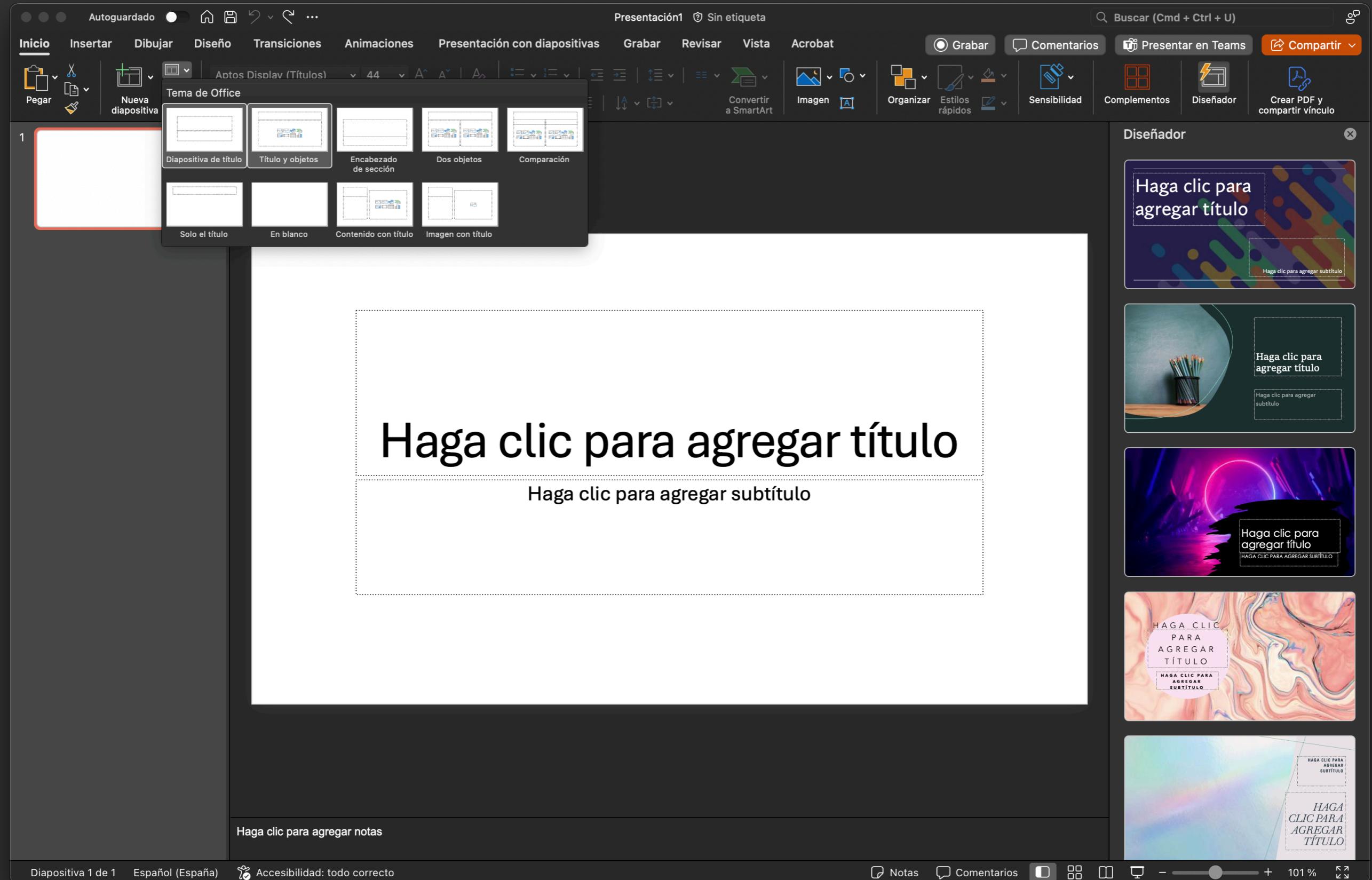
HAGA CLIC PARA AGREGAR TÍTULO

HAGA CLIC PARA AGREGAR SUBTÍTULO

HAGA CLIC PARA AGREGAR TÍTULO

HAGA CLIC PARA AGREGAR SUBTÍTULO

Notas Comentarios 00 00 101% 🔍



Autoguardado ⌂ ⌄ ⌅ ⌆ ⌇ ⌈ ⌉ ⌊ ⌋ ... Presentación1 Sin etiqueta Buscar (Cmd + Ctrl + U) ⌓

Inicio Insertar Dibujar Diseño Transiciones Animaciones Presentación con diapositivas Grabar Revisar Vista Acrobat Grabar Comentarios Presentar en Teams Compartir

Pegar Nueva diapositiva Aptos Display (Títulos) 44 A A A Convertir a SmartArt Imagen Organizar Estilos rápidos Sensibilidad Complementos Diseñador Crear PDF y compartir vínculo

N K S ab x x AV Aa A

1

Haga clic para agregar título

- Haga clic para agregar texto



Haga clic para agregar notas

Diseñador

Haga clic para agregar título
• Haga clic para agregar texto

Haga clic para agregar título
• Haga clic para agregar texto

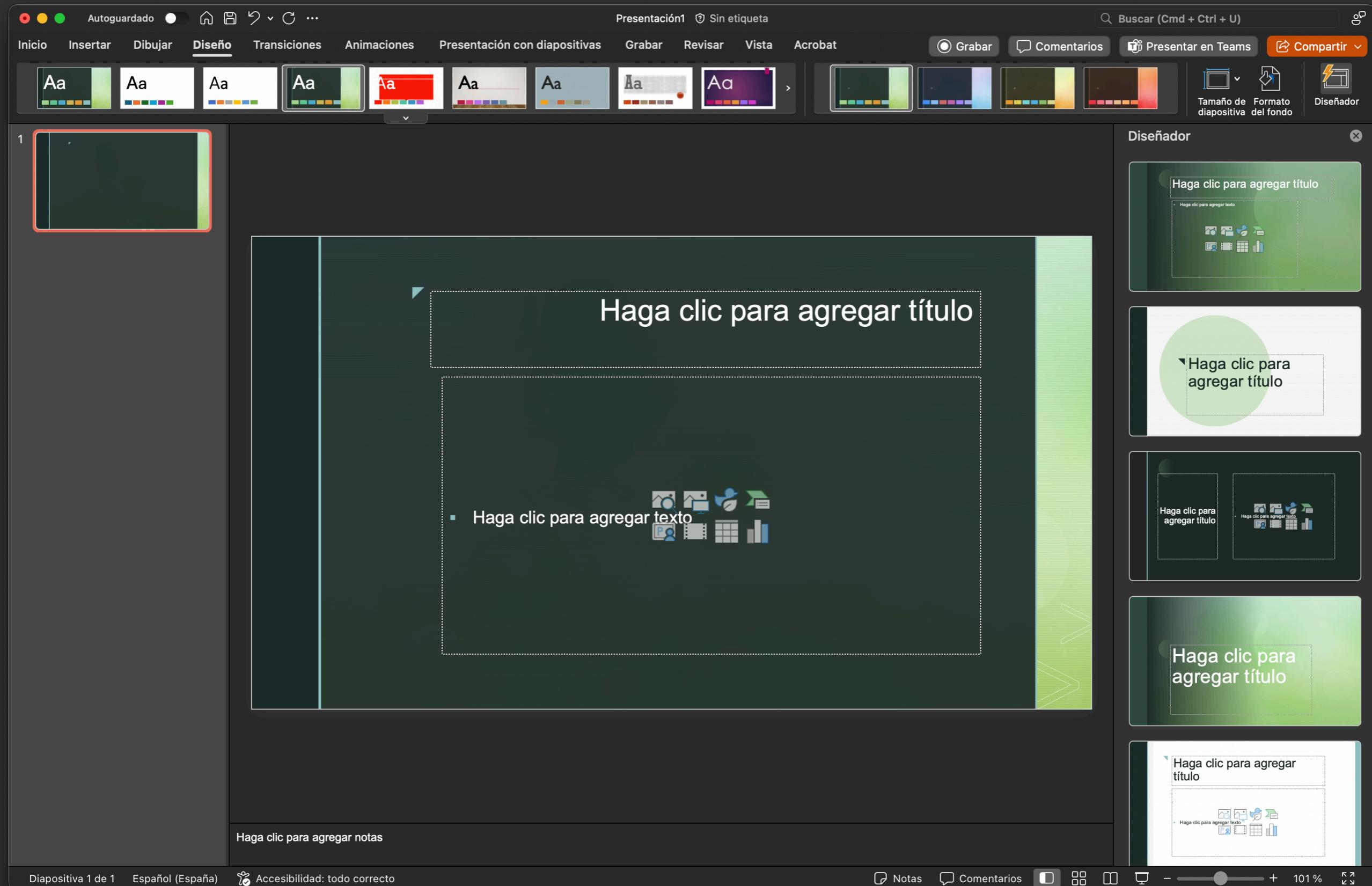
Haga clic para agregar título
• Haga clic para agregar texto

Haga clic para agregar título
• Haga clic para agregar texto

Haga clic para agregar título
• Haga clic para agregar texto

Haga clic para agregar título
• Haga clic para agregar texto

Diapositiva 1 de 1 Español (España) Accesibilidad: todo correcto Notas Comentarios ⌂ ⌃ ⌄ ⌅ ⌆ ⌇ ⌈ ⌉ ⌊ ⌋ - + 101% ⌓



Autoguardado

Presentación1 Sin etiqueta

Buscar (Cmd + Ctrl + U)

Inicio Insertar Dibujar **Diseño** Transiciones Animaciones Presentación con diapositivas Grabar Revisar Vista Acrobat Grabar Comentarios Presentar en Teams Compartir

Aa Aa

Tamaño de diapositiva Formato del fondo Diseñador

1

Haga clic para agregar título

Haga clic para agregar texto

Haga clic para agregar notas

Diseñador

Haga clic para agregar título

Haga clic para agregar texto

Haga clic para agregar título

Haga clic para agregar texto

Haga clic para agregar título

Haga clic para agregar texto

Haga clic para agregar título

Haga clic para agregar texto

Notas Comentarios 00 101% ↕

The screenshot shows a Microsoft PowerPoint presentation window. The main slide contains a red rectangular box with the placeholder text "Haga clic para agregar título". Below it is another red box with the placeholder "Haga clic para agregar texto". At the bottom of the slide, there is a text box with the placeholder "Haga clic para agregar notas". On the left side, there is a thumbnail of the first slide labeled "1". On the right side, there is a "Diseñador" sidebar containing five different design templates, each featuring a red box with the same placeholder text. The top navigation bar includes tabs like Inicio, Insertar, Dibujar, Diseño, Transiciones, Animaciones, Presentación con diapositivas, Grabar, Revisar, Vista, Acrobat, and options for Grabar, Comentarios, Presentar en Teams, and Compartir. The status bar at the bottom shows "Diapositiva 1 de 1" and "Español (España)".

Autoguardado ⚡ 🔍 ⌛ ⌂ ⌃ ⌄ ⌅ ⌆ ⌇ ... Presentación1 🗝 Sin etiqueta Buscar (Cmd + Ctrl + U) ☰

Inicio Insertar Dibujar **Diseño** Transiciones Animaciones Presentación con diapositivas Grabar Revisar Vista Acrobat Grabar Comentarios Presentar en Teams Compartir

Aa Tamaño de diapositiva Formato del fondo Diseñador

1

The slide features a green header bar with the text 'Haga clic para agregar título'. Below it is a large green callout bubble containing the text 'Haga clic para agregar título' and several small blue icons: a magnifying glass, a computer monitor, a leaf, a person, a grid, and a bar chart. The main content area has a dotted border and contains the text 'Haga clic para agregar texto' with a list icon. At the bottom left, there is a note placeholder with the text 'Haga clic para agregar notas'.

Diseñador

The sidebar displays five different design variations for the slide, each featuring a green header bar and a green callout bubble with icons. The designs vary in the placement and orientation of the callout bubble and the overall layout of the slide elements.

Diapositiva 1 de 1 Español (España) 🔍 Accesibilidad: todo correcto Notas Comentarios ⌂ ⌃ ⌄ ⌅ ⌆ ⌇ - + 101% ⌈ ⌉

Este tipo de diseños permite que parezca que los usuarios crean nuevas «clases» sin programar.

Como hicimos en el ejemplo de Omnigraffle.

Especificar nuevos objetos variando su estructura

A partir de partes y subpartes.

Los editores para el diseño de circuitos, por ejemplo, construyen circuitos utilizando subcircuitos. Por comodidad, estas aplicaciones suelen permitir instanciar estructuras complejas definidas por el usuario, por ejemplo, para utilizar un subcircuito concreto una y otra vez.

Lo mismo que hace Omnigraffle con las figuras definidas por el usuario.

Reduce las subclases

A diferencia del Factory Method, no hace falta una jerarquía paralela de creadores.

Su principal inconveniente es:

La necesidad de implementar la operación de clonación en todas las clases.

No siempre es fácil.

Especialmente si hace falta una copia profunda de los objetos, o en presencia de referencias circulares.

Implementación

Usar un gestor de prototipos

Cuando los prototipos no son fijos, sino que se pueden crear y eliminar dinámicamente, se puede utilizar un registro de los prototipos existentes.

Un gestor o registro de prototipos no es más que un diccionario que devuelve el prototipo apropiado para una clave dada.

Los clientes pueden así extender el sistema sin tocar el código.

Implementar la operación de clonación

Particularmente difícil si puede haber referencias circulares.

¿Copia profunda o superficial?

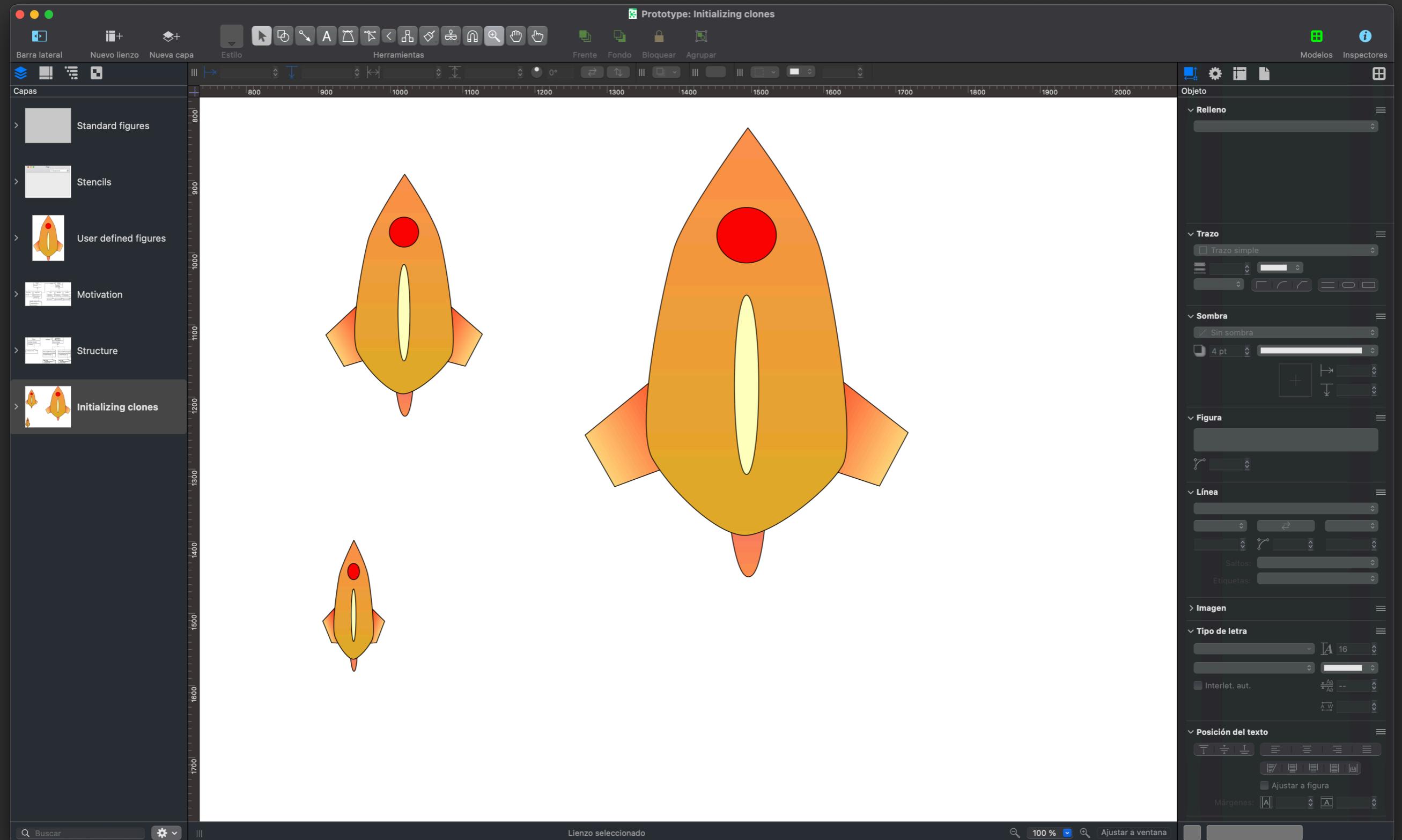
¿La clonación de un objeto debe clonar a su vez sus atributos, o basta con copiar sus referencias?

Inicializar los objetos clonados

Aunque a veces baste con una copia tal cual, otras los clientes querrán inicializar los objetos clonados con unos valores determinados.

No siempre es posible (o conveniente) pasar dichos valores en el propio método de clonación, en caso de que el número y tipo de parámetros pueda variar de unos prototipos a otros.

Otra posibilidad, si eso ocurre, es llamar a los métodos «set» correspondientes (o a algún tipo de operación de inicialización) justo tras la clonación.





CreationTool.java

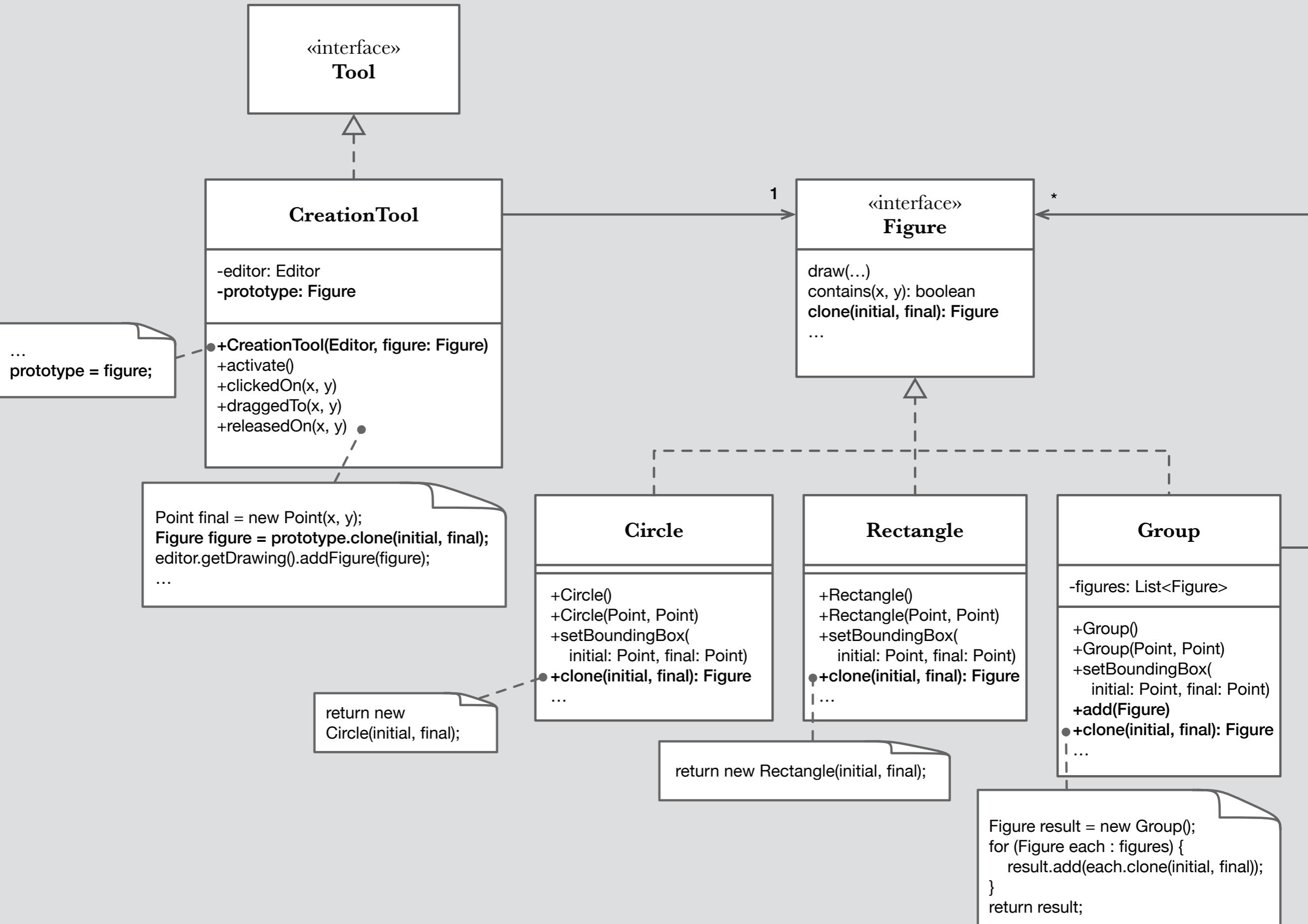
```
public class CreationTool implements Tool {  
    private Figure prototype;  
  
    public CreationTool(Figure prototype) {  
        this.prototype = prototype;  
    }  
  
    // ...  
  
    @Override  
    public void mouseUp(int x, int y) {  
        Point finalPoint = new Point(x, y);  
        Figure newFigure = prototype.clone(initialPoint,  
                                            finalPoint);  
        drawing.addFigure(newFigure);  
        editor.toolDone();  
    }  
}
```

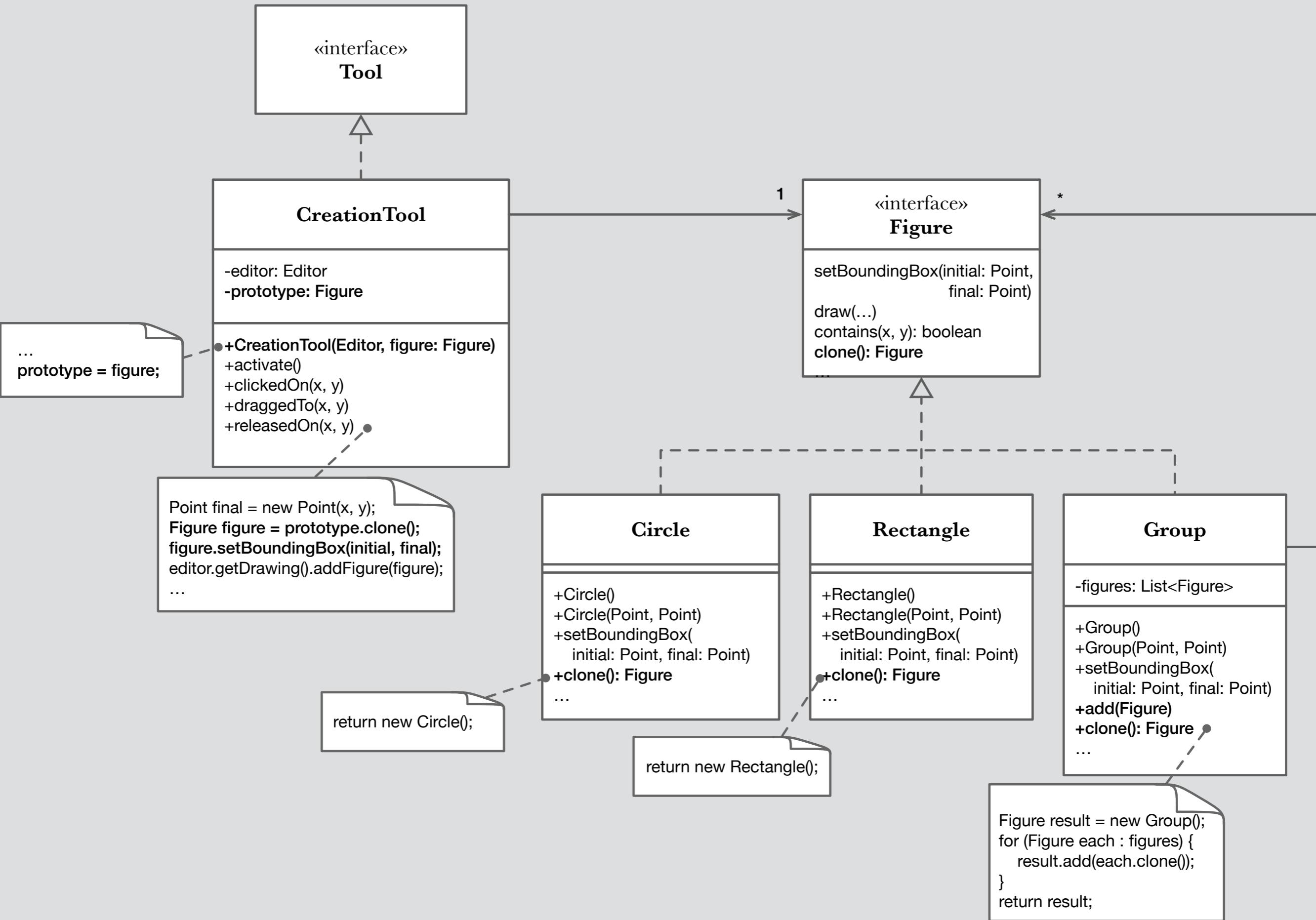


CreationTool.java

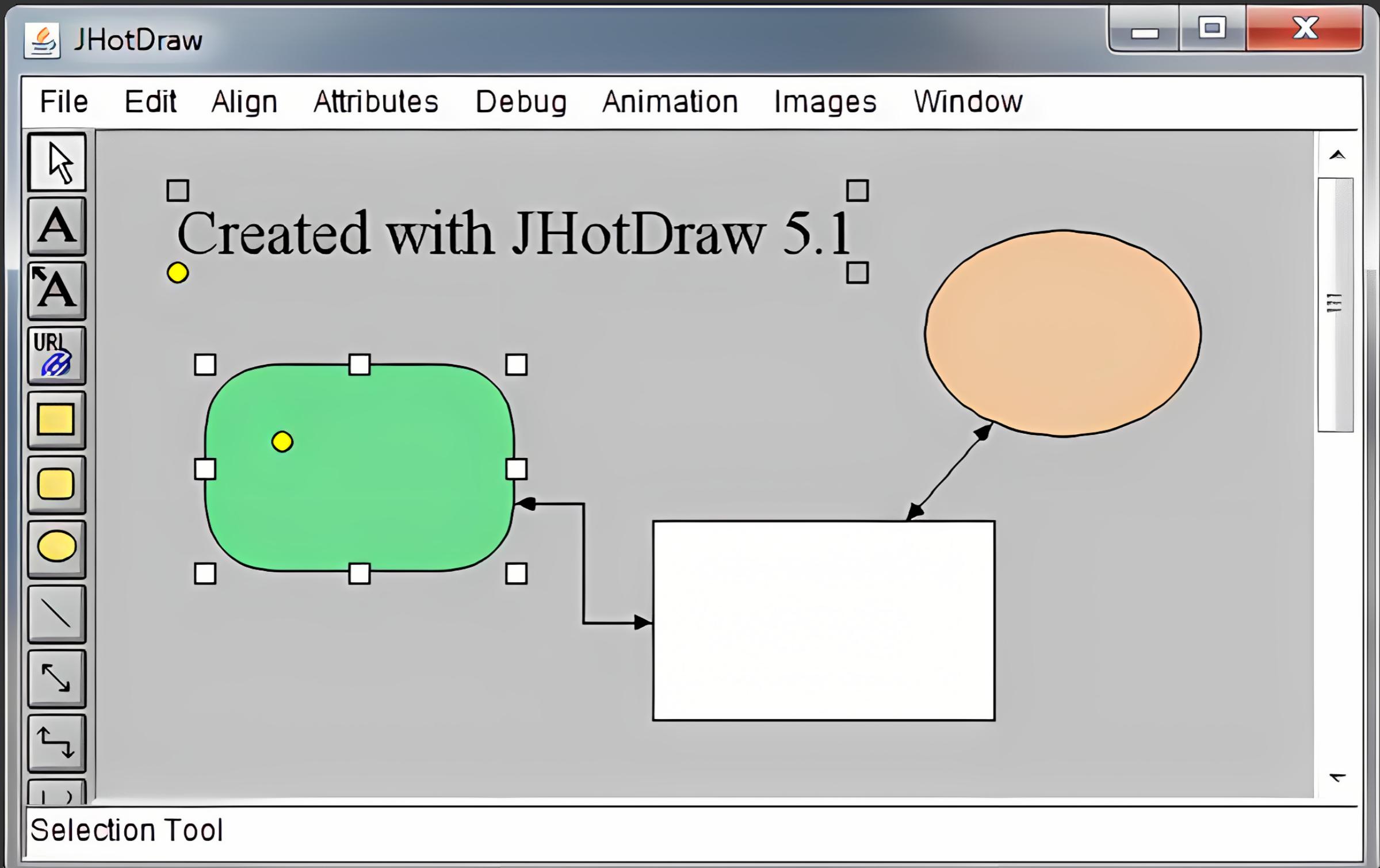
```
public class CreationTool implements Tool {  
    private Figure prototype;  
  
    public CreationTool(Figure prototype) {  
        this.prototype = prototype;  
    }  
  
    // ...  
  
    @Override  
    public void mouseUp(int x, int y) {  
        Point finalPoint = new Point(x, y);  
        Figure newFigure = prototype.clone();  
        figure.setBoundingBox(initialPoint, finalPoint);  
        drawing.addFigure(newFigure);  
        editor.toolDone();  
    }  
}
```

Código de ejemplo





Usos conocidos



Patrones relacionados

El Prototype y el Factory Method son, en muchos casos, patrones alternativos.

Por ese motivo, el Abstract Factory se puede implementar con una serie de prototipos, en vez de con varios Factory Method.