Escuela Colombiana de Ingeniería Julio Garavito  
Nicolas Cárdenas  
Paola Cuellar  
Laboratorio 01

**SHAPES**

El diagrama de clases permite visualizar las clases de un artefacto software y las relaciones entre ellas. Considerando el diagrama de clases de “shapes” ¿qué clases ofrece? ¿Qué relaciones existen entre ellas?

Ofrece las clases Canvas, Circle, Rectangle y Triangle  
Cada una de las clases está relacionadas al canvas, ya que cada uno hace uso de este recurso

De acuerdo con la documentación generada (visible desde su navegador): ¿qué clases tiene el paquete shapes? ¿Qué atributos tiene la clase Rectangle? ¿Cuáles métodos ofrece la clase Rectangle para que la figura cambie (incluya sólo el nombre)?

El paquete Shapes tiene la clase de Canvas, Circle, Rectangle y Triangle.  
Según la documentación generada no se puede ver los atributos de la clase  
Los métodos que ofrece son:   
changeColor  
changeSize  
makeInvisible  
makeVisible  
moveDown  
moveHorizontal  
moveLeft  
moveRight  
moveUp  
moveVertical  
slowMoveHorizontal  
slowMoveVertical

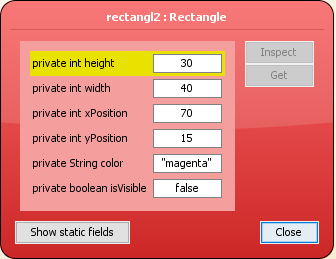
En el código de cada clase está el detalle de la implementación. Revisen el código de la clase Rectangle. Con respecto a los atributos: ¿cuántos atributos tiene? ¿Cuáles son privados y cuáles públicos? ¿Por qué? Con respecto a los métodos: ¿cuántos métodos tiene en total? ¿Cuáles son privados? ¿Quiénes los usan?

La clase rectangle tiene 6 atributos.  
Los atributos privados son: height, width, xPosition, yPosition, color, isVisible porque se realiza el encapsulamiento de los atributos para que el usuario no pueda realizar cambios en el **comportamiento** del objeto y no tiene atributos públicos.  
Tiene en total 15 métodos  
Los métodos privados son draw y erase y lo utilizan changeColor, changeSize, moveHorizontal, moveVertical, slowMoveHorizontal, slowMoveVertical, makeVisible, makeInvisible.

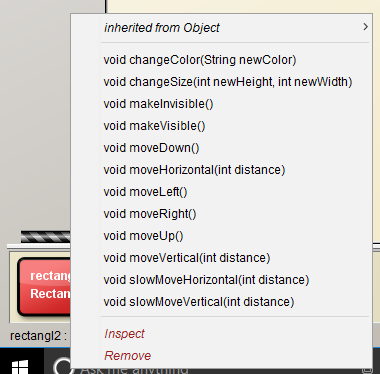
5. ¿Cuál dirían es el propósito del proyecto “shapes”?

A través de la programación ser capaces de mostrar uno o varios objetos (figuras geométricas como es el circulo, el rectángulo y el triángulo) de manera gráfica, permitiendo el cambio del comportamiento de los objetos para familiarizarnos con la herramienta.

**Manipulando objetos. Usando opciones.**  
1. Creen un objeto de cada una de las clases que lo permitan ¿Cuántas clases hay? ¿Cuántos objetos crearon? ¿Por qué?   
  
Hay 4 clases pero solo 3 de ellas son las que permiten la creación de objetos  
Se crearon 3 objetos, 1 por cada clase que lo permite porque la clase de Triangle, Circle y Rectangle tienen el constructor y el método de draw que permite la creación del objeto.

2. Inspeccione el estado del objeto Rectangle.

3. Inspeccione el comportamiento que ofrece el objeto Rectangle.



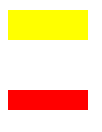
4. Construyan, con “shapes” sin escribir código, una propuesta de la imagen de su mascota favorita (incluyan una copia del original) ¿Cuántas y cuáles clases se necesitan? ¿Cuántos objetos se usan en total?



Se necesitaron el uso de una clase que fue Circle y Rectangle   
Se usaron 19 objetos de la clase Circle y 1 objeto de la clase Rectangle

**Manipulando objetos. Analizando y escribiendo código**

1. Revisen el código anterior e indiquen ¿Cuál es la figura resultante? Píntenla

La figura resultante es como la bandera de Colombia, pero sin la franja azul.

1. Habiliten la ventana de código en línea, escriban el siguiente código y en cada punto señalado indiquen: ¿Cuántas variables existen? ¿Cuántos objetos existen? ¿Cuántos objetos se ven? ¿Qué color tiene cada uno de ellos? Explique.



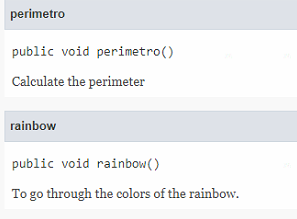
//1

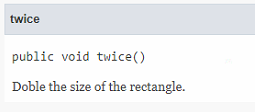
No existen variables  
No existen objetos  
No se ve ningún objeto   
No se puede determinar el color de un objeto que no existe

//2

Existen 3 variables  
Existen 2 objetos  
No se ve ningún objeto  
Tienen el color magenta por defecto en el constructor   
  
//3  
Existen 3 variables pero se está usando 1  
Existen 2 objetos pero se está modificando el comportamiento de 1  
Se ve 1 objeto   
El objeto que es actualmente visible es de color amarillo, el otro objeto sigue teniendo el color magenta por defecto del constructor  
  
//4  
Existen 3 variables pero se está usando 1  
Existen 2 objetos pero se está modificando el comportamiento de 1  
Se ven 2 objetos  
El primer objeto visible tiene el color amarillo, el segundo objeto visible tiene el color azul   
  
//5  
Existen 3 variables pero se está usando 1  
Existen 2 objetos pero se está modificando el comportamiento de 1  
Se ven 2 objetos  
El primer objeto visible tiene el color amarillo, el segundo objeto visible tiene el color azul   
  
//6  
Existen 3 variables pero se está usando 1  
Existen 2 objetos pero se está modificando el comportamiento de 1  
Se ven 2 objetos  
El primer objeto visible tiene el color amarillo, el segundo objeto visible tiene el color rojo   
  
3. Es igual la figura generada a la inicial ¿por qué?  
La figura generada en el punto 1 con respecto a la figura del punto 2, son las mismas; ya que en el punto 1 se escribe el código para hacer todas las acciones de forma directa y en el punto 2 se escribe el código en la ventana de código en línea que realiza las mismas acciones pero línea por línea, dejando ver el paso a paso de lo que se realiza y llega a la misma figura.

**Extendiendo clases**1. Método rainbow()  
  
2. Método twice()  
  
3. Método perimetro()  
  
4. Generar nuevamente la documentación y revise la información de estos nuevos métodos





**Implementando una nueva clase. Molecule.**

Métodos constructores  
 \_(type : String) : Molecule

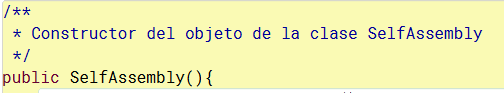
Métodos analizadores  
 getType() : String

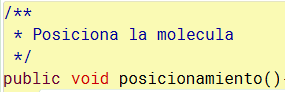
Métodos modificadores

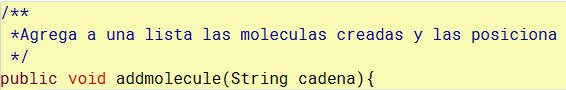
makeVisible() : void  
makeInvisible() : void

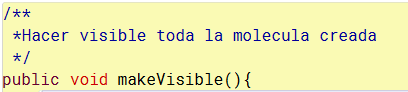
moveDown() : void  
moveRight() : void  
reflect() : void  
rotate() : void  
  
 **Definiendo y creando una nueva clase. SelfAssembly**

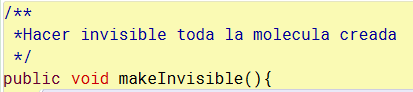
1.

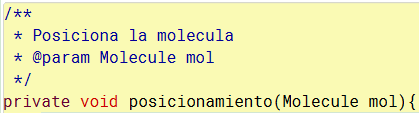


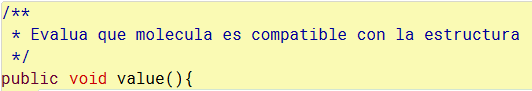


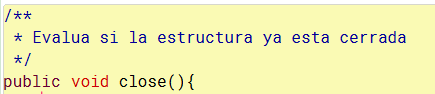












**RETROSPECTIVA**

1. Nicolas Cardenas

Paola Cuellar

2. El estado actual del laboratorio es incompleto, porque durante el desarrollo del mismo tuvimos dudas sobre las variaciones que podía tener en el momento del desarrollo del SelfAssembly

3. La práctica más útil fue trabajar con un compañero en un mismo computador porque compartimos ideas y puntos de vistas que permiten el desarrollo de este

4. El mayor logro fue ponernos de acuerdo en el modo que abordaríamos el desarrollo del problema