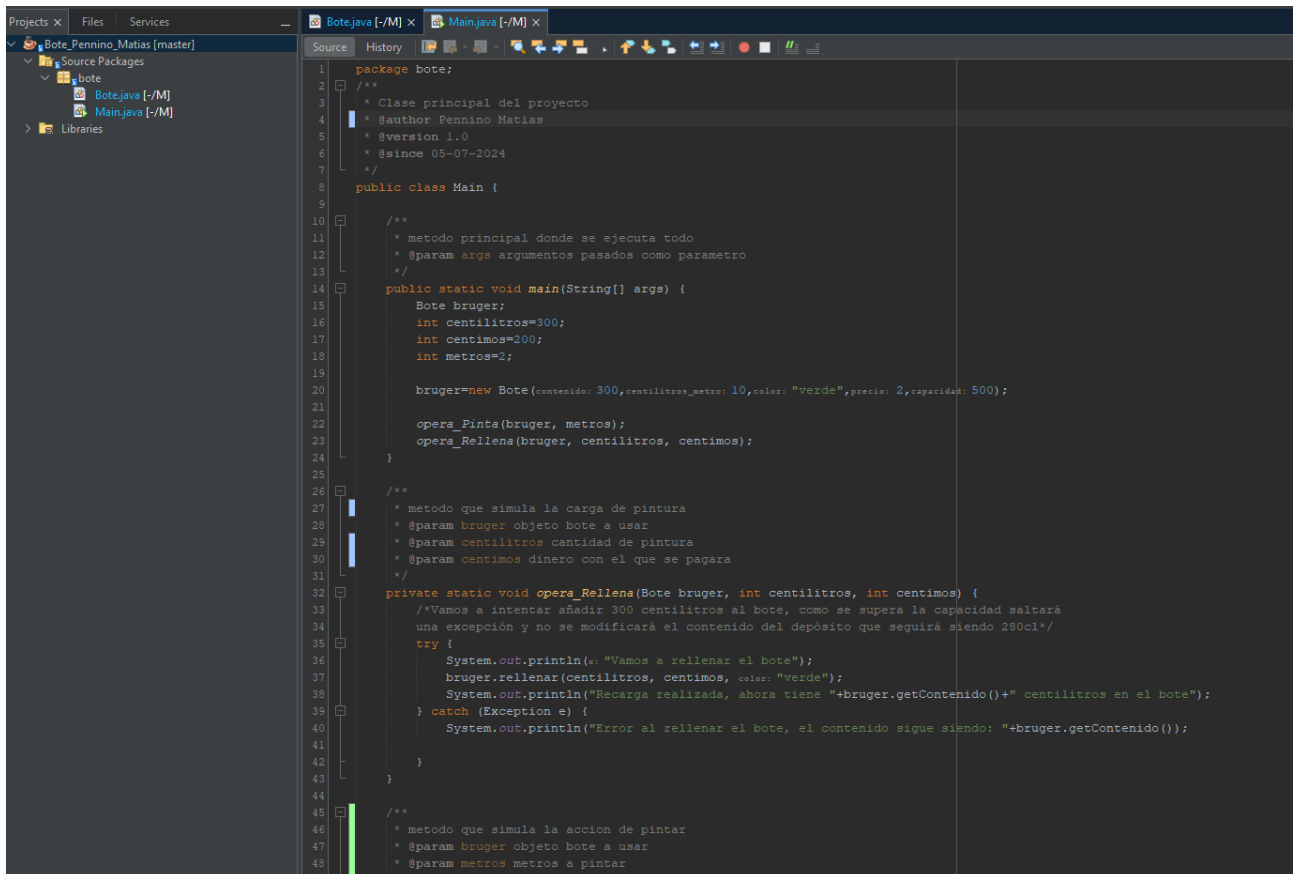


Actividad 4.4.2 Documentación javadoc proyecto Bote

Alumno: Matias Pennino



```
1 package bote;
2 /**
3  * Clase principal del proyecto
4  * @author Pennino Matias
5  * @version 1.0
6  * @since 05-07-2024
7  */
8 public class Main {
9
10     /**
11     * metodo principal donde se ejecuta todo
12     * @param args argumentos pasados como parametro
13     */
14     public static void main(String[] args) {
15         Bote bruger;
16         int centilitros=300;
17         int centimos=200;
18         int metros=2;
19
20         bruger=new Bote(contenido: 300,centilitros_metro: 10,color: "Verde",precio: 2,capacidad: 500);
21
22         opera_Pinta(bruger, metros);
23         opera_Rellena(bruger, centilitros, centimos);
24     }
25
26     /**
27     * metodo que simula la carga de pintura
28     * @param bruger objeto bote a usar
29     * @param centilitros cantidad de pintura
30     * @param centimos dinero con el que se pagara
31     */
32     private static void opera_Rellena(Bote bruger, int centilitros, int centimos) {
33         /**Vamos a intentar añadir 300 centilitros al bote, como se supera la capacidad saltará
34         una excepción y no se modificará el contenido del depósito que seguirá siendo 280cl*/
35         try {
36             System.out.println("Vamos a rellenar el bote");
37             bruger.rellenar(centilitros, centimos, color: "Verde");
38             System.out.println("Recarga realizada, ahora tiene "+bruger.getContenido()+" centilitros en el bote");
39         } catch (Exception e) {
40             System.out.println("Error al rellenar el bote, el contenido sigue siendo: "+bruger.getContenido());
41         }
42     }
43
44     /**
45     * metodo que simula la accion de pintar
46     * @param bruger objeto bote a usar
47     * @param metros metros a pintar
48     */
49 }
```

```
25
26
27 /**
28  * metodo que simula la carga de pintura
29  * @param bruger objeto bote a usar
30  * @param centilitros cantidad de pintura
31  * @param centimos dinero con el que se pagara
32  */
33 private static void opera_Rellena(Bote bruger, int centilitros, int centimos) {
34     /*Vamos a intentar añadir 300 centilitros al bote, como se supera la capacidad saltará
35     una excepción y no se modificará el contenido del depósito que seguirá siendo 280cl*/
36     try {
37         System.out.println("Vamos a rellenar el bote");
38         bruger.rellenar(centilitros, centimos, color: "verde");
39         System.out.println("Recarga realizada, ahora tiene "+bruger.getContenido()+" centilitros en el bote");
40     } catch (Exception e) {
41         System.out.println("Error al rellenar el bote, el contenido sigue siendo: "+bruger.getContenido());
42     }
43 }
44
45 /**
46  * metodo que simula la accion de pintar
47  * @param bruger objeto bote a usar
48  * @param metros metros a pintar
49  */
50 private static void opera_Pinta(Bote bruger, int metros) {
51     /*Vamos a pintar 2 metros, como hay suficiente pintura la operación tendrá éxito y se
52     descontarán 20 centilitros del contenido del bote, quedan 280cl*/
53     try {
54         System.out.println("Vamos a pintar");
55         bruger.pintar(metros);
56         System.out.println("En el bote quedan "+bruger.getContenido()+" centilitros");
57     } catch (Exception e) {
58         System.out.println("Error al pintar");
59     }
60 }
61
62 }
```

Capturas de la clase Main

```
49
50 /**
51  * metodo que devuelve el valor del bote
52  * @return precio del bote
53  */
54 public int getPrecio() {
55     return precio;
56 }
57
58 /**
59  * metodo que permite establecer el valor del bote
60  * @param precio nuevo precio del bote
61  */
62 public void setPrecio(int precio) {
63     this.precio = precio;
64 }
65
66 /**
67  * metodo que devuelve la capacidad del bote
68  * @return la capacidad del bote
69  */
70 public int getCapacidad() {
71     return capacidad;
72 }
73
74 /**
75  * metodo que permite establecer la cantidad del bote
76  * @param capacidad capacidad maxima del bote
77  */
78 public void setCapacidad(int capacidad) {
79     this.capacidad = capacidad;
80 }
81
82 private int contenido; //Centilitros de pintura disponibles en el bote
83 private int centilitros_metro; //Centilitros necesarios para pintar un metro cuadrado
84 private String color; //Color de la pintura
85 private int precio; //Precio en céntimos del centilitro de pintura
86 private int capacidad; //Número máximo de centilitros que caben en el bote
87
88 /**
89  * constructor sin parametros
90  */
91 public Bote() {}
92
93 /**
94  * constructor con 5 parametros
95  * @param contenido cuanta pintura tiene el bote
96  * @param centilitros_metro cuantos centilitros consume por metro
97  * @param color color de la pintura
98  * @param precio valor del bote
99  * @param capacidad capacidad maxima del bote
100 */
```

```
91  /**
92   * constructor con 5 parametros
93   * @param contenido cuanta pintura tiene el bote
94   * @param centilitros_metro cuantos centilitros consume por metro
95   * @param color color de la pintura
96   * @param precio valor del bote
97   * @param capacidad capacidad maxima del bote
98   */
99  public Bote(int contenido, int centilitros_metro, String color, int precio, int capacidad) {
100      this.contenido=contenido;
101      this.centilitros_metro=centilitros_metro;
102      this.color=color;
103      this.precio=precio;
104      this.capacidad=capacidad;
105  }
106
107  /**
108   * metodo que devuelve el contenido en centilitros del bote
109   * @return contenido del bote
110   */
111  public int getContenido() {
112      return contenido;
113  }
114
115
116
117  /**
118   * Método que permite pintar una superficie, es necesario que haya suficiente
119   * pintura en el bote y que la superficie que se va a pintar sea positiva
120   * @param metros metros a pintar
121   * @throws Exception si la pintura es insuficiente o los metros son negativos
122   */
123  public void pintar(int metros) throws Exception{
124      if(metros*this.getCentilitros_metro()>this.getContenido())
125          throw new Exception(message:"No hay pintura suficiente para pintar esa superficie");
126      if(metros<=0)
127          throw new Exception(message:"No tiene sentido pintar una superficie que no sea positiva");
128      this.setContenido(this.getContenido() - (metros * this.getCentilitros_metro()));
129  }
130
131
132  /**
133   * Método que permite aumentar el contenido del bote de pintura, es necesario
134   * que la cantidad que se quiere añadir y el dinero con que se va a pagar
135   * sean positivos, además el dinero debe ser suficiente para pagar la cantidad
136   * que se va a añadir
137   * @param centilitros cantidad de pintura a rellenar
138   * @param centimos dinero con el que se pagara
139   * @param color color de pintura
140   * @throws Exception si el dinero o los centilitros son negativos o 0, si se
```

```
128      this.setContenido(this.getContenido() - (metros * this.getCentilitros_metro()));
129  }
130
131  /**
132   * Método que permite aumentar el contenido del bote de pintura, es necesario
133   * que la cantidad que se quiere añadir y el dinero con que se va a pagar
134   * sean positivos, además el dinero debe ser suficiente para pagar la cantidad
135   * que se va a añadir
136   * @param centilitros cantidad de pintura a rellenar
137   * @param centimos dinero con el que se pagara
138   * @param color color de pintura
139   * @throws Exception si el dinero o los centilitros son negativos o 0, si se
140   * quiere llenar mas de lo posible el bote, o si el dinero es insuficiente
141   */
142  public void rellenar(int centilitros, int centimos, String color) throws Exception{
143      if (centimos<=0){
144          throw new Exception(message:"Se necesita una cantidad positiva de dinero");
145      }
146      if (centilitros<=0){
147          throw new Exception(message:"Los centilitros deben ser positivos");
148      }
149
150      if (this.getContenido() + centilitros > getCapacidad()) {
151          throw new Exception(message:"No se puede superar la capacidad del bote");
152      }
153      if (centimos < centilitros * getPrecio()){
154          throw new Exception(message:"No tiene dinero suficiente");
155      }
156      this.setContenido(this.getContenido() + centilitros);
157  }
158
159  }
160
```

Capturas de la clase Bote