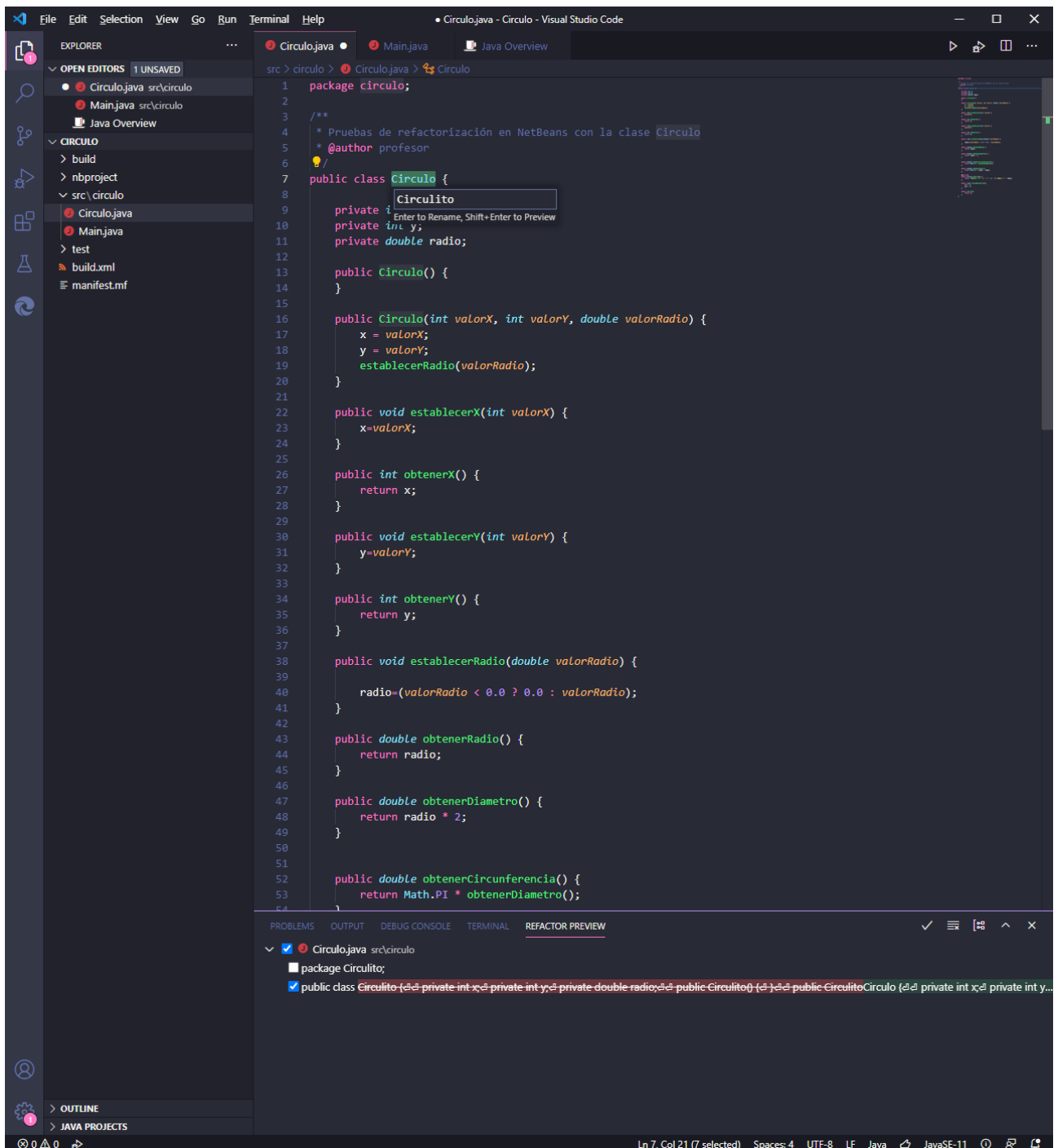
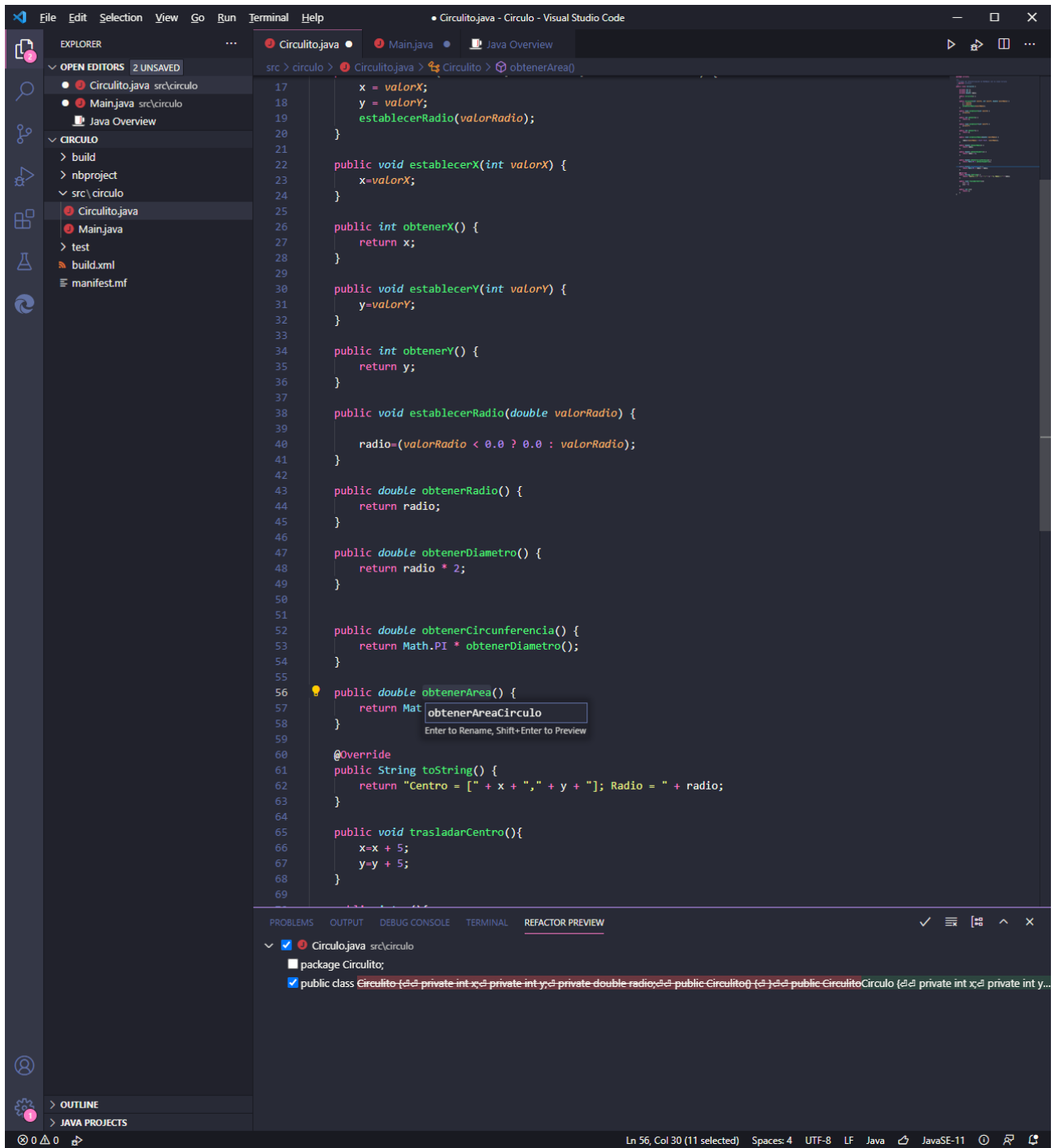


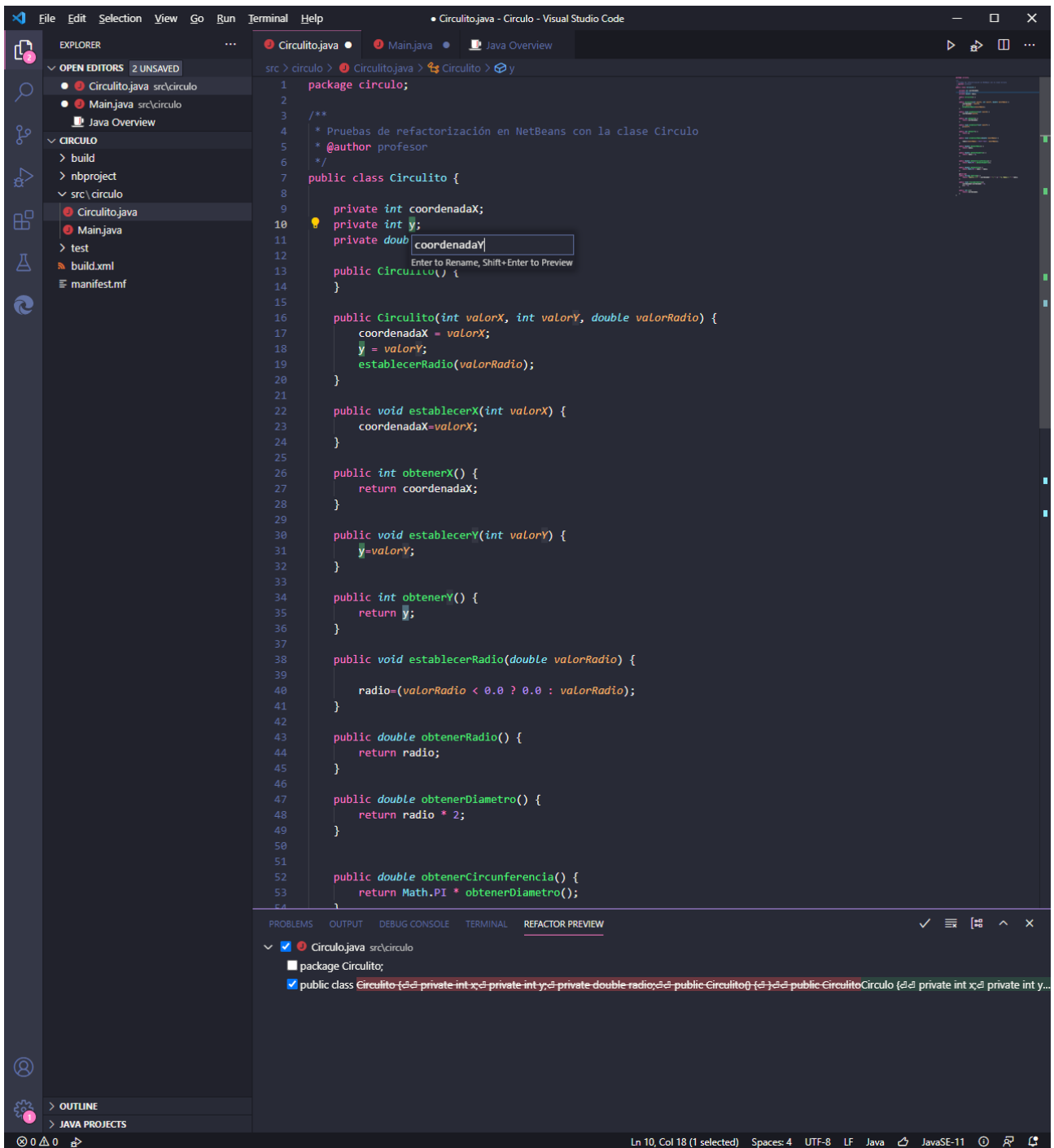
- Renomear a classe Circulo por Circulito.



- Renomear o método ObtenerArea por ObtenerAreaCirculo.



- Renomear os campo x e y por coordenadaX e coordenadaY.



- Introducir constante LIMITERADIO de tipo double co valor 0.0

The screenshot shows the Visual Studio Code editor with a Java project named 'Circulito'. The Explorer sidebar on the left shows the project structure with files like 'Circulito.java', 'Main.java', 'build.xml', and 'manifest.mf'. The main editor displays the code for 'Circulito.java'. The code defines a 'Circulito' class with several attributes: 'LIMITERADIO' (a static final double set to 0.0), 'coordenadaX', 'coordenadaY', and 'radio'. It includes methods for setting and getting these values, as well as methods to calculate the diameter, circumference, and area of the circle. A refactor preview window is open at the bottom, showing a suggestion to rename the 'radio' attribute to 'radioD'.

```
6 //
7 public class Circulito {
8
9     private static final double LIMITERADIO = 0.0;
10     private int coordenadaX;
11     private int coordenadaY;
12     private double radio;
13
14     public Circulito() {
15     }
16
17     public Circulito(int valorX, int valorY, double valorRadio) {
18         coordenadaX = valorX;
19         coordenadaY = valorY;
20         establecerRadio(valorRadio);
21     }
22
23     public void establecerX(int valorX) {
24         coordenadaX=valorX;
25     }
26
27     public int obtenerX() {
28         return coordenadaX;
29     }
30
31     public void establecerY(int valorY) {
32         coordenadaY=valorY;
33     }
34
35     public int obtenerY() {
36         return coordenadaY;
37     }
38
39     public void establecerRadio(double valorRadio) {
40
41         radio=(valorRadio < LIMITERADIO ? 0.0 : valorRadio);
42     }
43
44     public double obtenerRadio() {
45         return radio;
46     }
47
48     public double obtenerDiametro() {
49         return radio * 2;
50     }
51
52
53     public double obtenerCircunferencia() {
54         return Math.PI * obtenerDiametro();
55     }
56
57     public double obtenerArea() {
58         return Math.PI * radio * radio;
59     }
60 }
```

REFACTOR PREVIEW

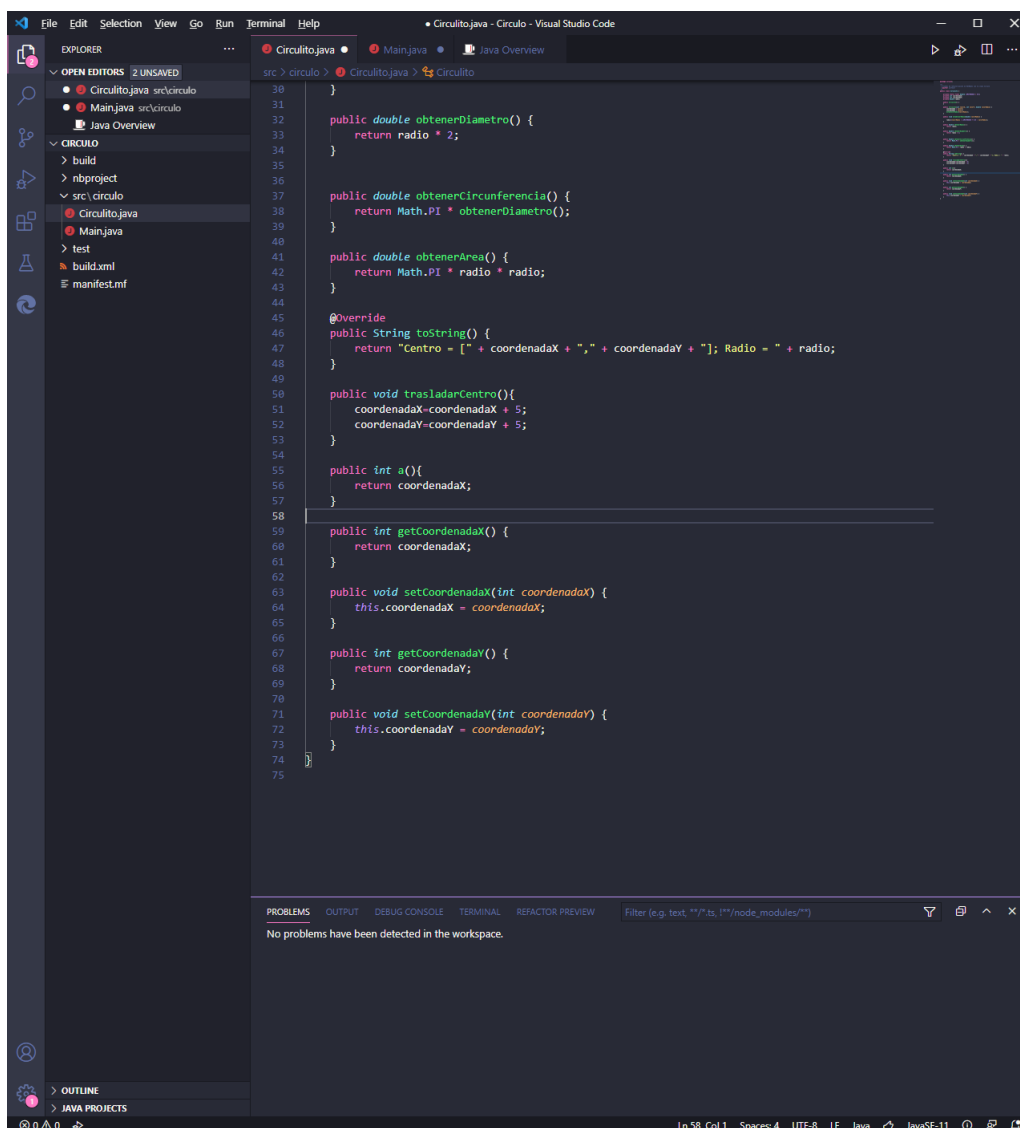
- ✓ ☒ package Circulito;
- ✓ ☒ public class Circulito {~~private int x;~~~~private int y;~~~~private double radio;~~~~public Circulito()~~~~public Circulito(int x, int y, double radio)~~~~public Circulito~~ public CirculitoCirculo (~~private int x;~~ private int y, double radio) {

Ln 41, Col 29 Spaces: 4 UTF-8 LF Java JavaSE-11

- Eliminar de forma segura os métodos obtenerX, obtenerY, establecerX, establecerY haciendo os cambios necesarios no código para que sexan substituídos polos correspondentes métodos tipo get e set creados.

Botón derecho en “establecercoordenadaX” y le pinchamos en “source action”, se nos abre una nueva ventana donde pinchamos en “generar getters and setter” y arriba pinchamos en los dos que solicitan. Se generan al final del documento, con lo que podemos borrar ya “establecercoordenadaX” y “establecercoordenadaY”.

Tenemos que cambiar también los datos en la clase main, para que este se pueda ejecutar.



The screenshot shows the Visual Studio Code interface with the file 'Circulito.java' open. The Explorer sidebar on the left shows the project structure with 'src/circulo' expanded. The main editor displays the code for 'Circulito' class. A right-click context menu is open over the 'establecercoordenadaX' method, showing options like 'Copy', 'Paste', 'Delete', and 'Source Action'. The 'Source Action' option is highlighted, and a submenu is visible with 'Generate Getters and Setters' selected. The code in the editor includes methods for diameter, circumference, area, toString, trasladarCentro, and getters/setters for coordenadaX and coordenadaY.

```
30 }
31
32 public double obtenerDiametro() {
33     return radio * 2;
34 }
35
36
37 public double obtenerCircunferencia() {
38     return Math.PI * obtenerDiametro();
39 }
40
41 public double obtenerArea() {
42     return Math.PI * radio * radio;
43 }
44
45
46 @Override
47 public String toString() {
48     return "Centro = [" + coordenadaX + ", " + coordenadaY + "]; Radio = " + radio;
49 }
50
51 public void trasladarCentro(){
52     coordenadaX=coordenadaX + 5;
53     coordenadaY=coordenadaY + 5;
54 }
55
56 public int a(){
57     return coordenadaX;
58 }
59
60 public int getCoordenadaX() {
61     return coordenadaX;
62 }
63
64 public void setCoordenadaX(int coordenadaX) {
65     this.coordenadaX = coordenadaX;
66 }
67
68 public int getCoordenadaY() {
69     return coordenadaY;
70 }
71
72 public void setCoordenadaY(int coordenadaY) {
73     this.coordenadaY = coordenadaY;
74 }
75 }
```

- **Optativo.** Encapsular os tres campos do método (coordenadaX coordenadaY, radio). Investigar a funcionalidade de encapsular.

Xera métodos get e set para un campo e opcionalmente actualiza tódalas referencias a ese campo utilizando os métodos get e set

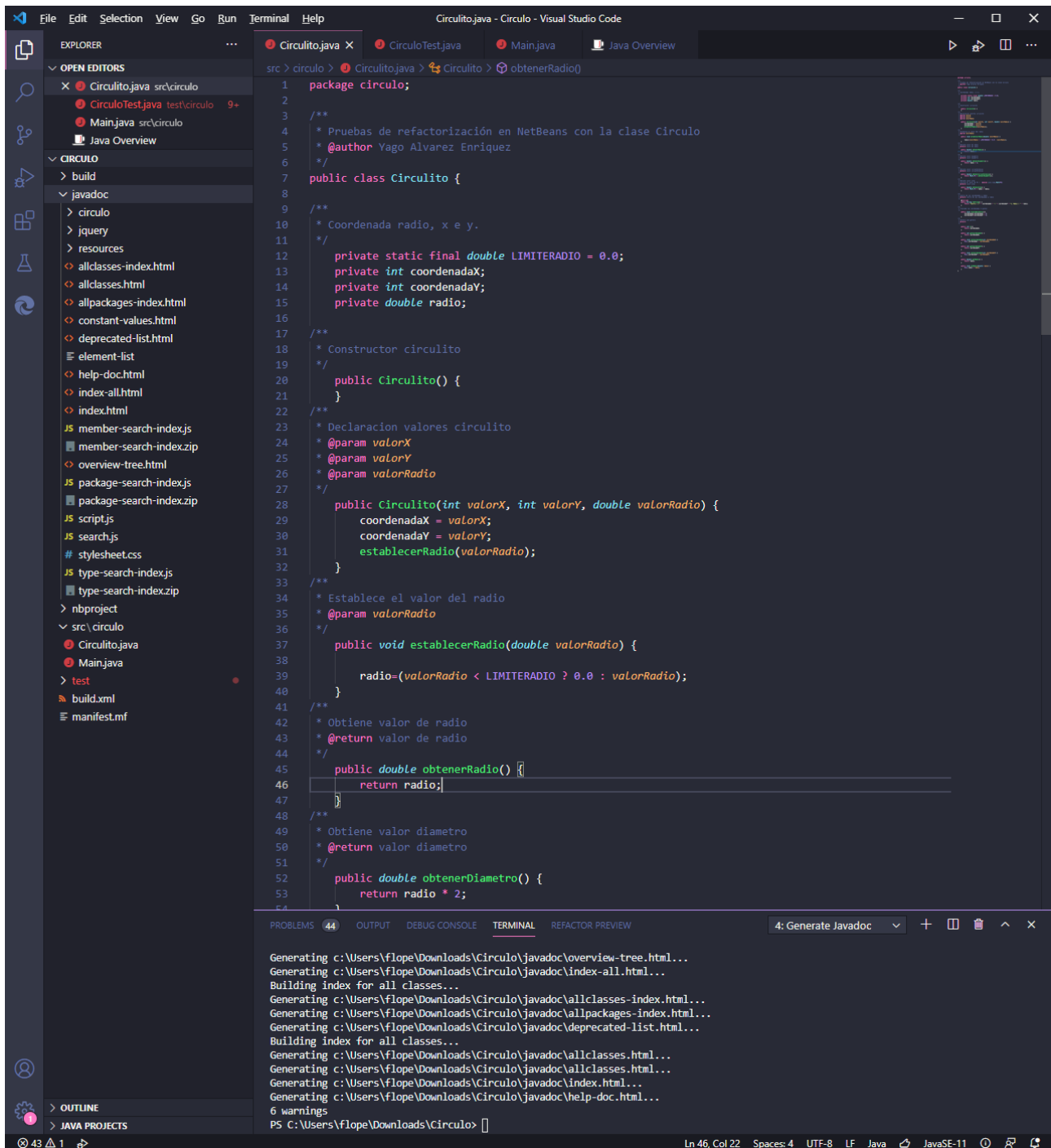
```

1 package circulo;
2
3 /**
4  * Pruebas de refactorización en NetBeans con la clase Circulo
5  * @author profesor
6  */
7 public class Circulito {
8
9     private static final double LIMITERADIO = 0.0;
10    private int coordenadaX;
11    private int coordenadaY;
12    private double radio;
13
14    public Circulito() {
15    }
16
17    public Circulito(
18        coordenadaX
19        coordenadaY
20        establecerRadio) {
21    }
22
23    public void establecerRadio(double valorRadio) {
24
25        radio=(valorRadio < LIMITERADIO ? 0.0 : valorRadio);
26    }
27
28    public double obtenerRadio() {
29        return radio;
30    }
31
32    public double obtenerDiametro() {
33        return radio * 2;
34    }
35
36
37    public double obtenerCircunferencia() {
38        return Math.PI * obtenerDiametro();
39    }
40
41    public double obtenerArea() {
42        return Math.PI * radio * radio;
43    }
44
45    @Override
46    public String toString() {
47        return "Centro = [" + coordenadaX + "," + coordenadaY + "]; Radio = " + radio;
48    }
49
50    public void trasladarCentro(){
51        coordenadaX=coordenadaX + 5;
52        coordenadaY=coordenadaY + 5;
53    }
54

```

Visual Studio Code interface showing the refactoring of the 'radio' field in the 'Circulito' class. The context menu is open, showing options like 'Generate Getters and Setters...'. The code shows the 'Circulito' class with private fields 'coordenadaX', 'coordenadaY', and 'radio', and public methods for getting and setting them.

Exercicio 2. Javadoc. Documenta o proxecto Círculo. Usa todas as etiquetas que consideres necesario. Xera a páxina web asociada ao Javadoc. (No caso de que che de erro adxunta unha imaxe do erro)

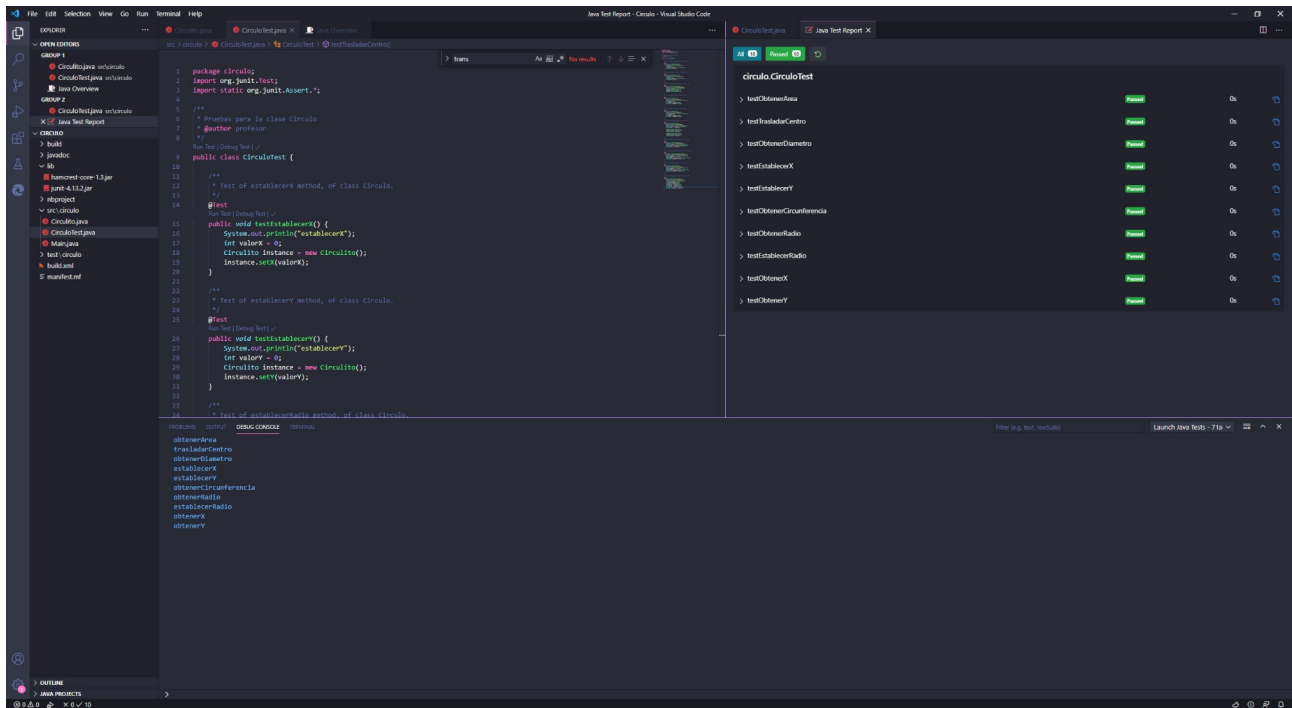


The screenshot displays the Visual Studio Code interface with the 'Círculo.java' file open in the editor. The file contains the following Java code:

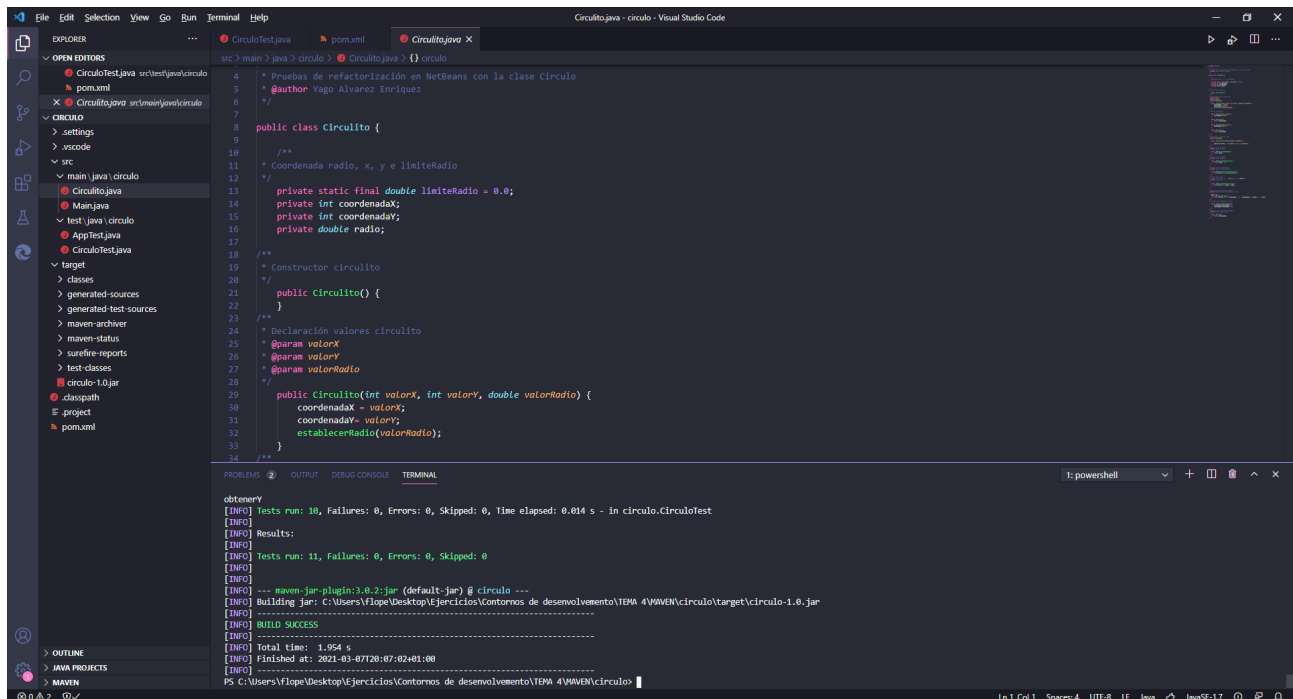
```
1 package círculo;
2
3 /**
4  * Pruebas de refactorización en NetBeans con la clase Círculo
5  * @author Yago Alvarez Enríquez
6  */
7 public class Círculo {
8
9  /**
10   * Coordenada radio, x e y.
11   */
12   private static final double LIMITERADIO = 0.0;
13   private int coordenadaX;
14   private int coordenadaY;
15   private double radio;
16
17  /**
18   * Constructor círculo
19   */
20   public Círculo() {
21   }
22
23  /**
24   * Declaracion valores círculo
25   * @param valorX
26   * @param valorY
27   * @param valorRadio
28   */
29   public Círculo(int valorX, int valorY, double valorRadio) {
30       coordenadaX = valorX;
31       coordenadaY = valorY;
32       establecerRadio(valorRadio);
33   }
34
35  /**
36   * Establece el valor del radio
37   * @param valorRadio
38   */
39   public void establecerRadio(double valorRadio) {
40       radio=(valorRadio < LIMITERADIO ? 0.0 : valorRadio);
41   }
42
43  /**
44   * Obtiene valor de radio
45   * @return valor de radio
46   */
47   public double obtenerRadio() {
48       return radio;
49   }
50
51  /**
52   * Obtiene valor diametro
53   * @return valor diametro
54   */
55   public double obtenerDiametro() {
56       return radio * 2;
57   }
58 }
```

The Explorer view on the left shows the project structure, including the 'javadoc' directory. The Terminal view at the bottom shows the output of the '4: Generate Javadoc' command, indicating the successful generation of Javadoc files in the 'c:\Users\flope\Downloads\Círculo\javadoc' directory.

Adjunto captura de todos los test pasados ya que me dieron muchos problemas. Todo era por no tener el hamcrest T__T.

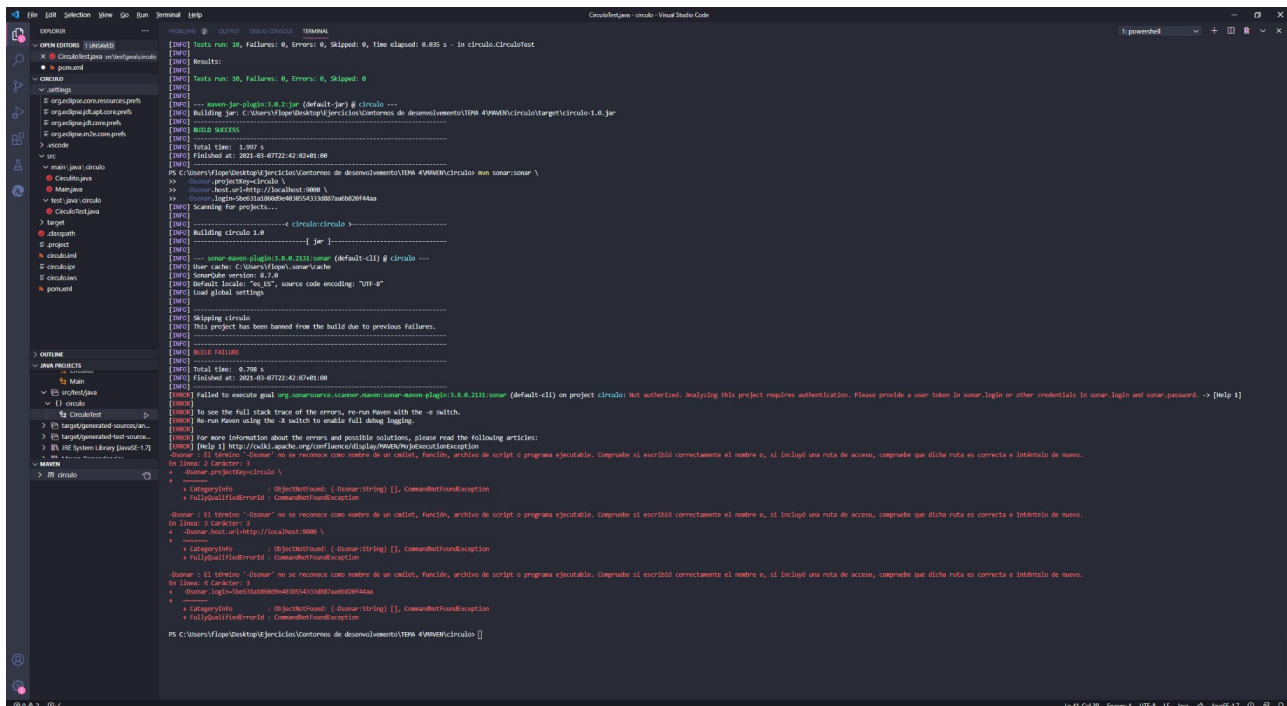


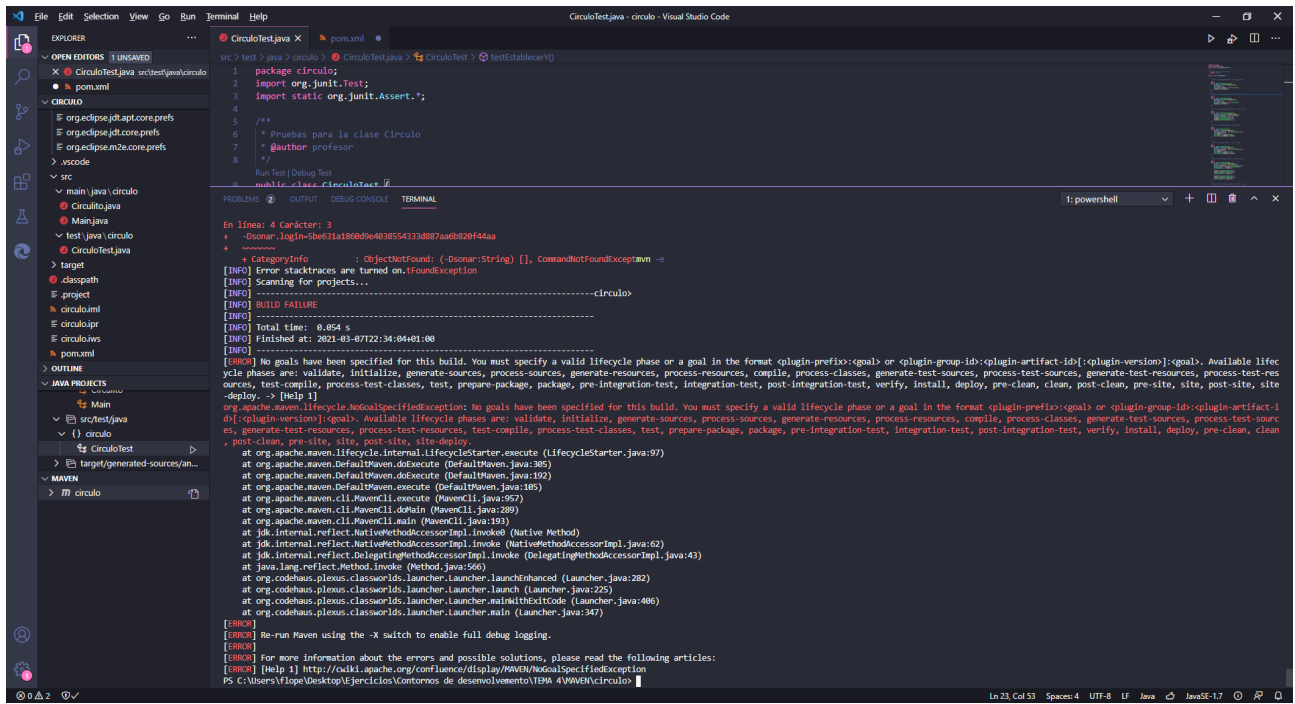
Exercicio 3. SonarQube. Instala o software SonarQube e analiza a calidade do proxecto Circulo.



El nvm clean package me funcionaba perfectamente.

A partir de aquí he intentado durante toda la tarde subir los datos a sonarqube y no he sido capaz, me salta siempre un error. Ha sido bastante desesperante no avanzar en todo el día de este error, intenté con tu proyecto virgen, con otros, y ha sido imposible. He probado todos los comandos posibles que he visto en internet, pero ni así.





Analyze your project

We initialized your project on SonarQube, now it's up to you to launch analyses!

1 Provide a token

circulo-5be631a1860d9e4038554333d887aa6b820f44aa

2 Run analysis on your project

What is your build technology?

Maven ☐ **Gradle** ☐ **.NET** ☐ **Other (for JS, TS, Go, Python, PHP, ...)** ☐

Execute the Scanner for Maven from your computer

Running a SonarQube analysis with Maven is straightforward. You just need to run the following command in your project's folder.

```
mvn sonar:sonar \
  -Dsonar.projectKey=circulo \
  -Dsonar.host.url=http://localhost:9000 \
  -Dsonar.login=5be631a1860d9e4038554333d887aa6b820f44aa
```

Copy

Please visit the [official documentation of the Scanner for Maven](#) for more details.

Once the analysis is completed, this page will automatically refresh and you will be able to browse the analysis results.

He probado con `nvm -X` y me lleva esta web: [NoGoalSpecifiedException - Apache Maven - Apache Software Foundation](#) , pero tampoco consigo nada.

Te dejo las variables por si el error viene de aquí, que tampoco lo sé.

