# Implementación de la solución

# para el empaquetado de SCORM

Por: Andrés Felipe Diaz G.

Proyecto aplicado en TIC[[1]](#footnote-1)

Universidad Pontificia Bolivariana

Medellín

2020

Contenido

[Implementación de la solución 1](#_Toc40715064)

[para el empaquetado de SCORM 1](#_Toc40715065)

[Empaquetado SCORM 3](#_Toc40715066)

[Objetos importantes 3](#_Toc40715067)

[Implementación en Java 4](#_Toc40715068)

[Clase ConexionSCORM 4](#_Toc40715069)

[Constructor 5](#_Toc40715070)

[Método inicio de clase 5](#_Toc40715071)

[Método guardar actividades 6](#_Toc40715072)

[Método para utilizar las constantes 6](#_Toc40715073)

[Método librería Zip4j 8](#_Toc40715074)

[Clase interna Content 8](#_Toc40715075)

[El constructor 10](#_Toc40715076)

[Clase interna Actividad 11](#_Toc40715077)

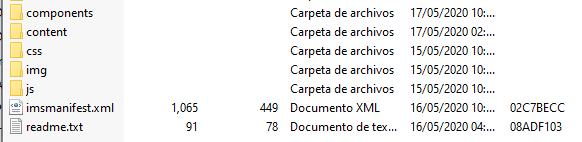
[Clase Recurso 11](#_Toc40715078)

[Cibergrafía 12](#_Toc40715079)

# Empaquetado SCORM

Como nos pudimos dar cuenta en el documento de investigación a cerca de una forma para realizar el empaquetamiento tipo SCORM llegamos a la conclusión de que la mejor opción era realizar un empaquetamiento usando una plantilla y modificando los aspectos necesarios para el funcionamiento.

Veamos de que está constituida la plantilla:



Tenemos estas carpetas en las cuales se almacenan archivos necesarios para que el empaquetado quede bajo los estándares de SCORM.

En cada una de ellas se encuentras diferentes archivos, en su mayoría con extensión .css que se encargan de agregar las componentes necesarias para considerar nuestro -zip final uno de tipo e-learning.

Ahora bien, y es preciso mencionarlo, en nuestro readme.txt no tenemos una explicación como la que hay en este documento, tenemos un requisito muy importante y es que los archivos que sean material de clase deberán guardarse en la carpeta “content” que se ve en la imagen anterior.

## Objetos importantes

Mencionemos también un detalle bastante importante, en la carpeta de js se encuentran todos los JavaScripts que se encargan de hacer las conexiones y posibilitar la comunicación entre el imsmanifest[[2]](#footnote-2) y los demás archivos necesarios como el index.html.

## Implementación en Java

Para lo anterior creamos una clase que se va a llamar “*ConexionSCORM”* la cual nos va a generar los archivos html necesarios para realizar la actividad por medio de una plataforma LMS.

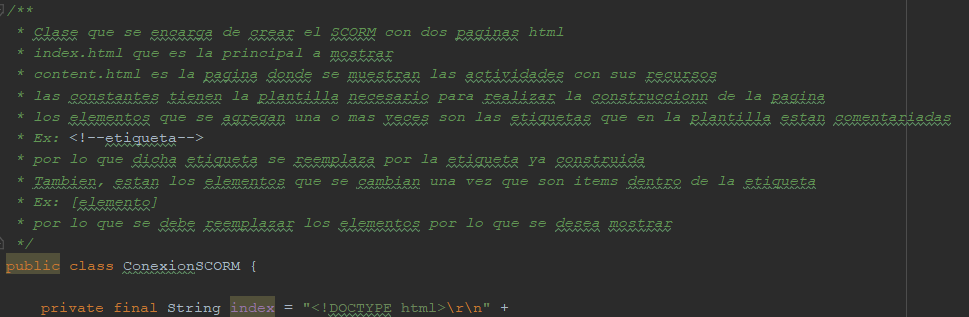
Entre los anteriores es bueno mencionar el que podría ser el más importante de todos, el imsmanifest.xml



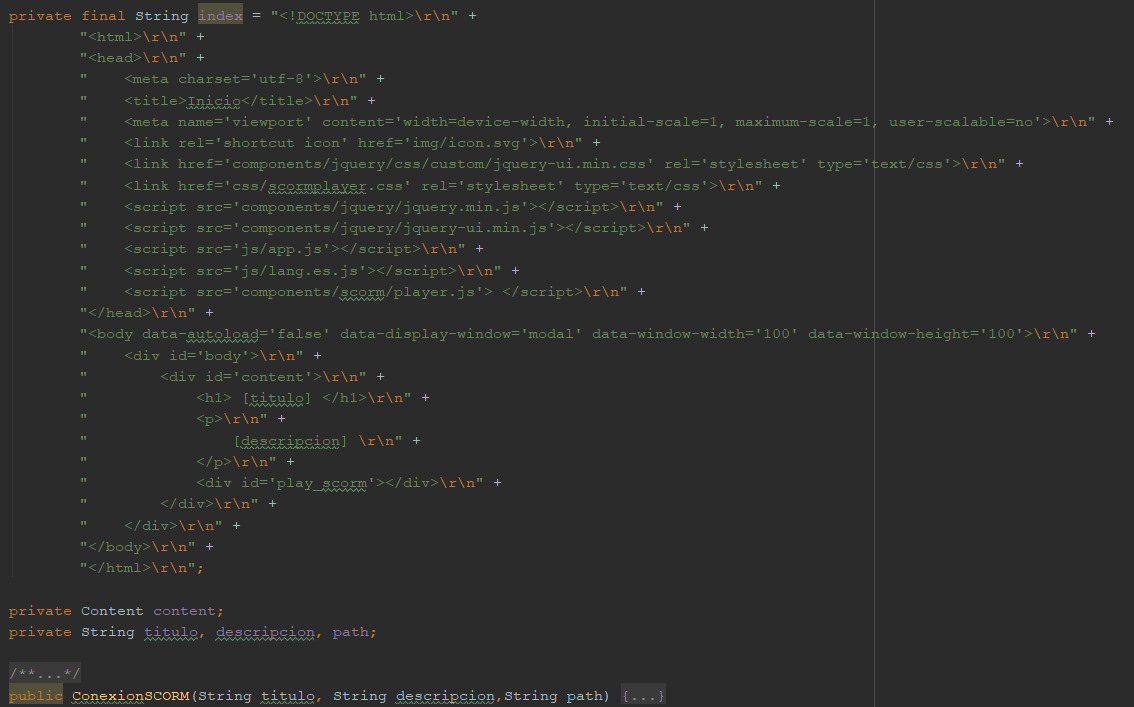
Así se ve el archivo imanifest.xml adaptado a nuestros requerimientos.

## Clase ConexionSCORM

Dicha clase utiliza una estructura básica con el imanifest mencionado anteriormente que nos permite crear paquetes SCORM 1.2 como se muestra a continuación

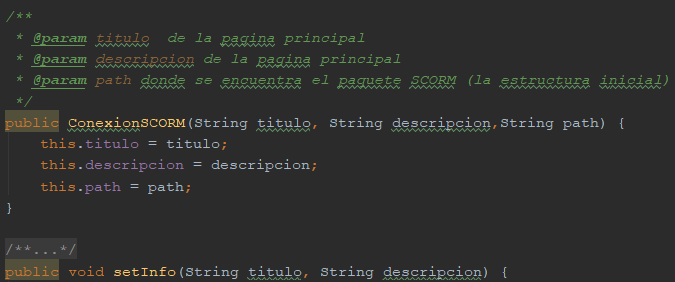


La plantilla de las paginas están en las mismas variables constantes que se utilizan para reemplazar las partes donde va la información



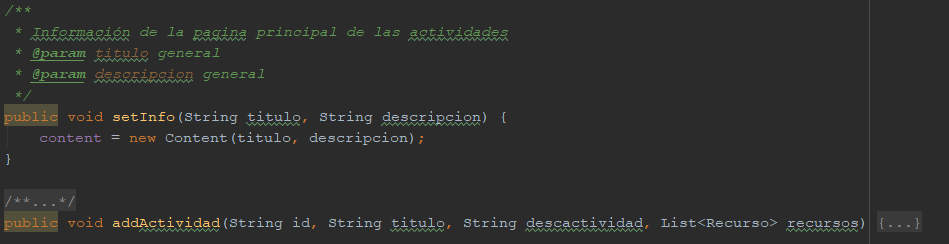
### Constructor

El constructor de la clase que establece la información de la página principal, que es la general de la sesión y nos muestra un botón para iniciar todas las actividades:



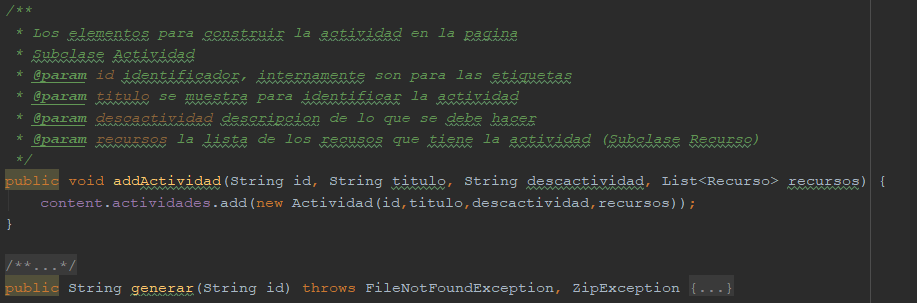
### Método inicio de clase

Luego, está el método para iniciar la clase que maneja la estructura de la página que muestra todas las actividades a realizar:



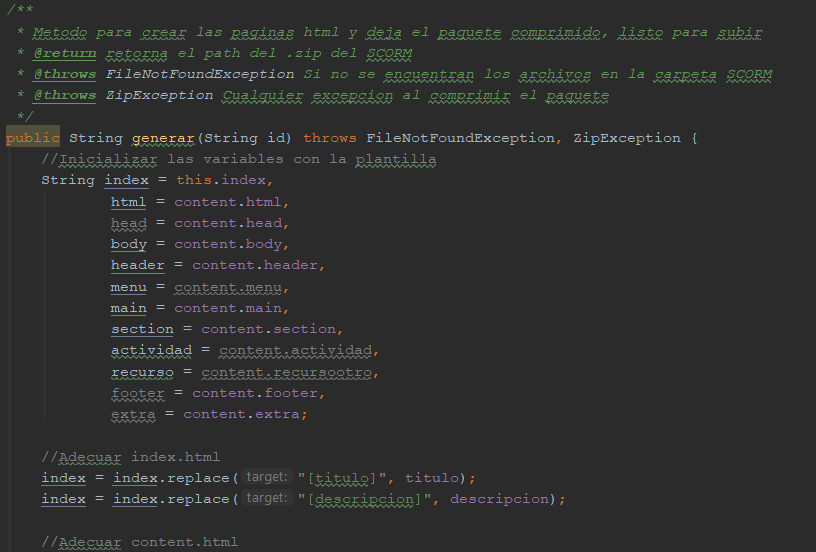
### Método guardar actividades

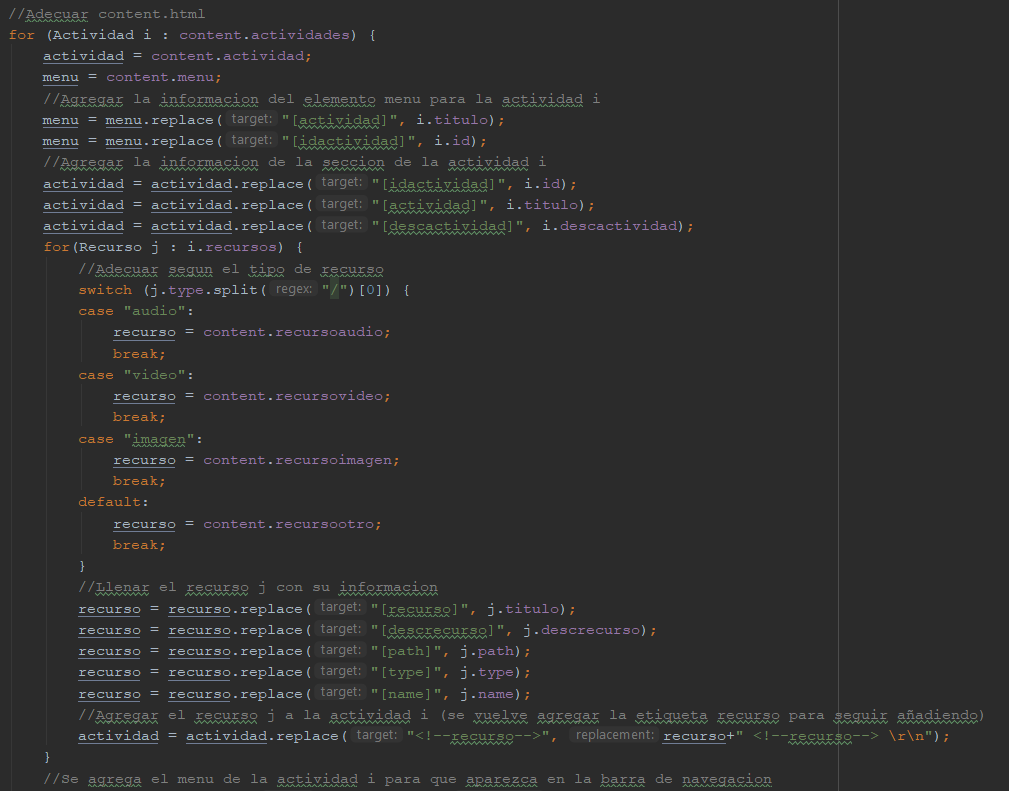
Después, el método que guarda las actividades con sus recursos que se mostraran:

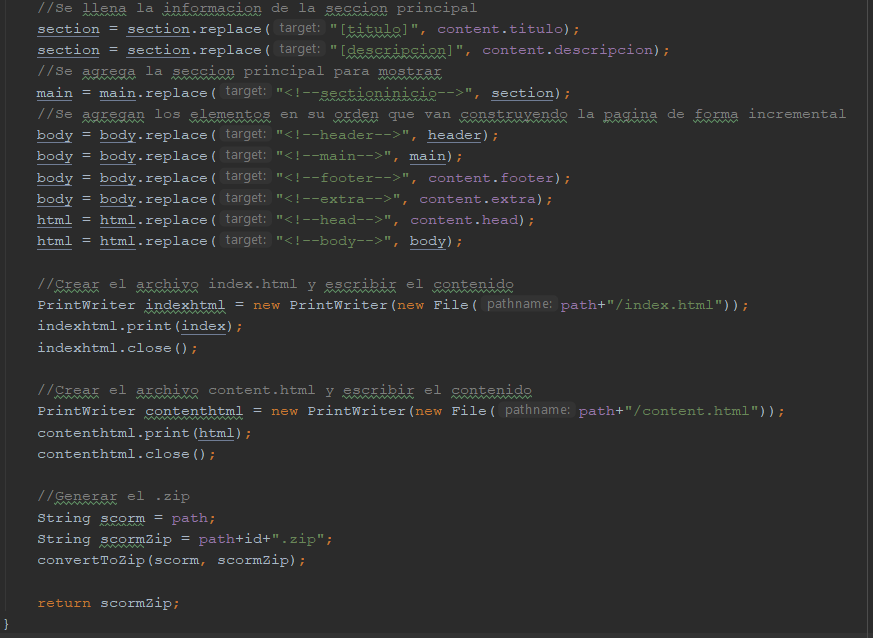


### Método para utilizar las constantes

También, está el método que se encarga de utilizar las constantes para crear las páginas con toda la información suministrada, además genera el paquete SCORM.zip para ser utilizado en las plataformas LMS

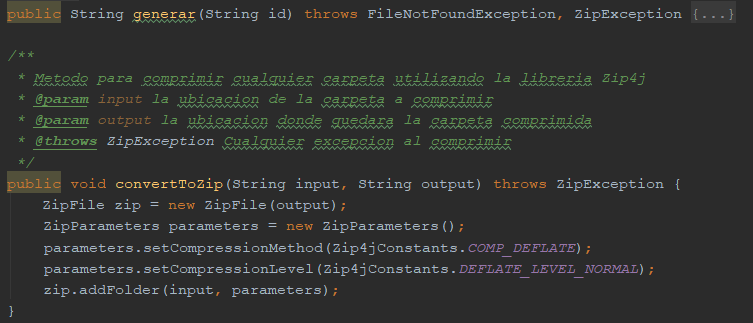






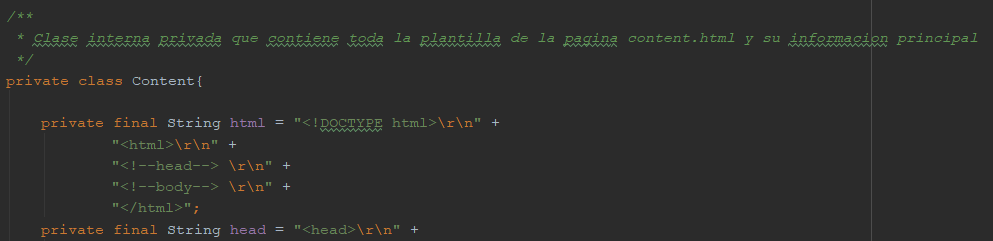
### Método librería Zip4j

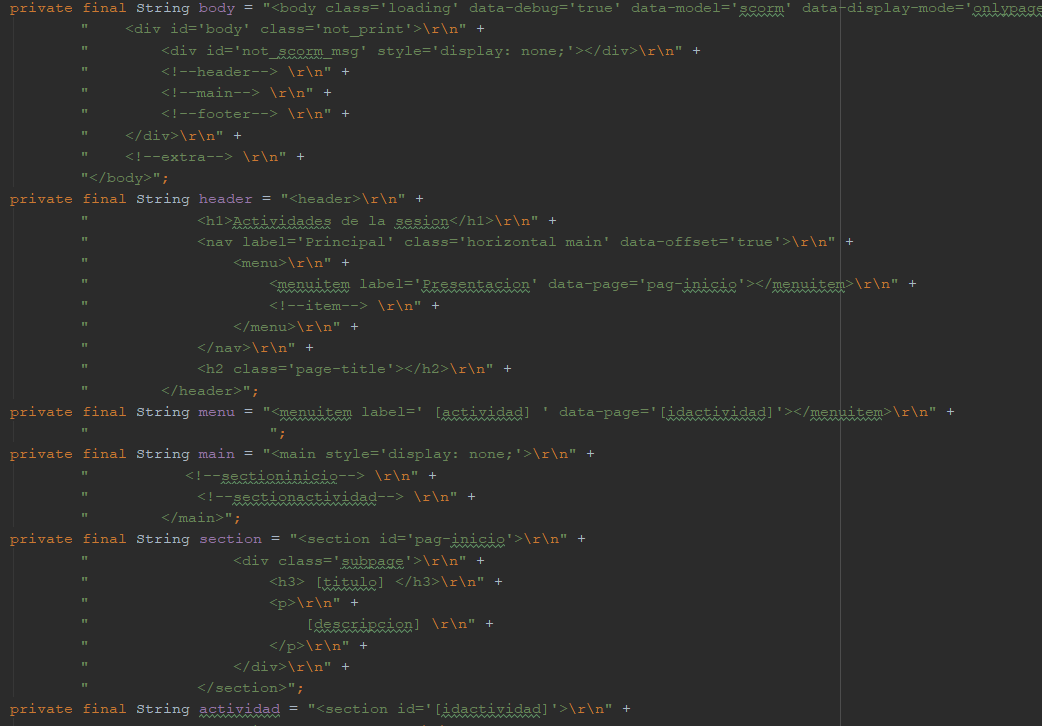
Y el método para generar el comprimido utilizando la librería Zip4j en la versión 1.3.2.

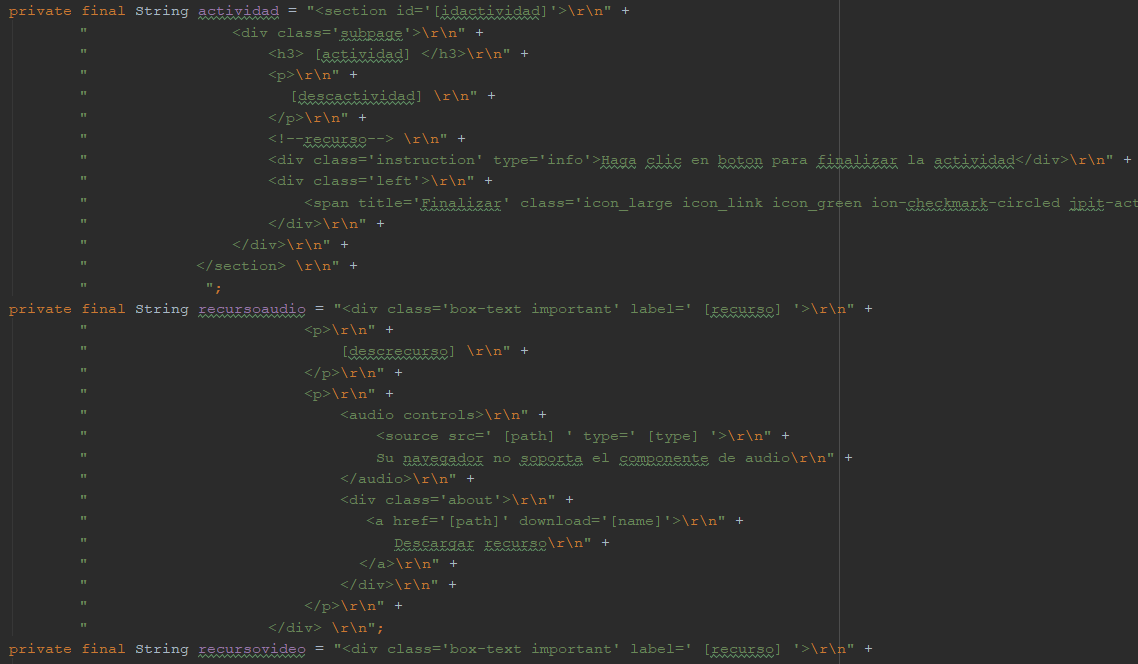


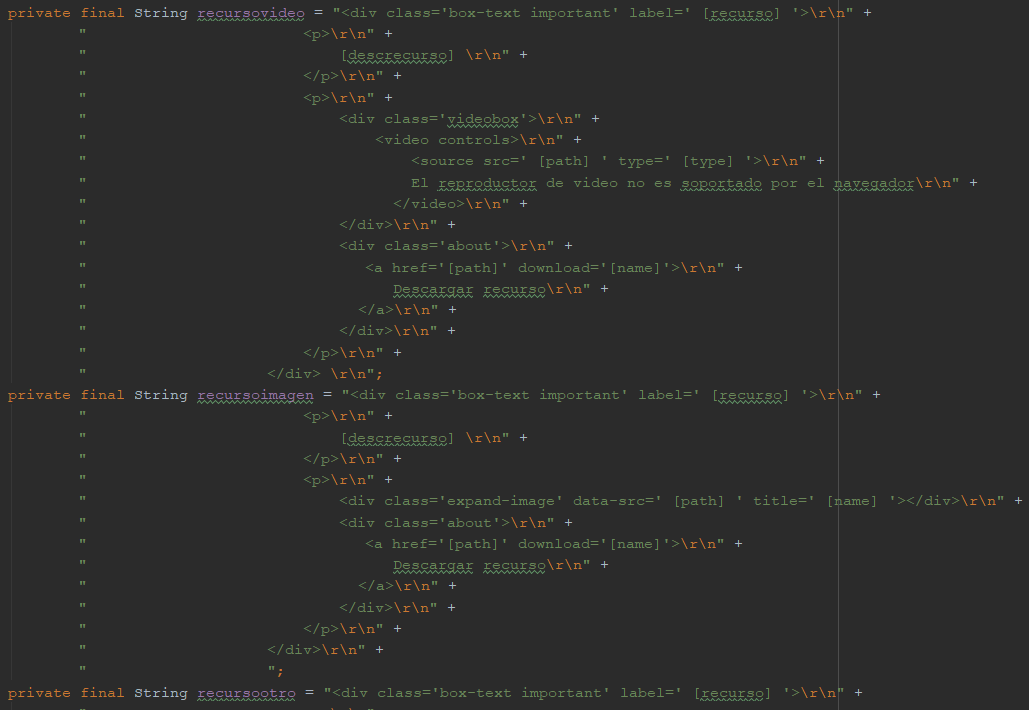
## Clase interna Content

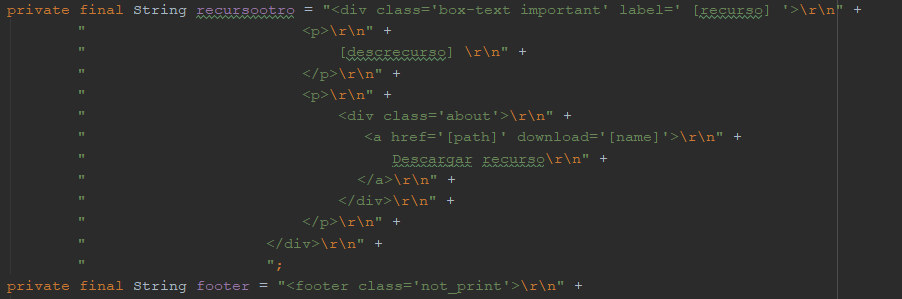
Además, está la clase interna *Content* que contiene la plantilla de la página conten.html:

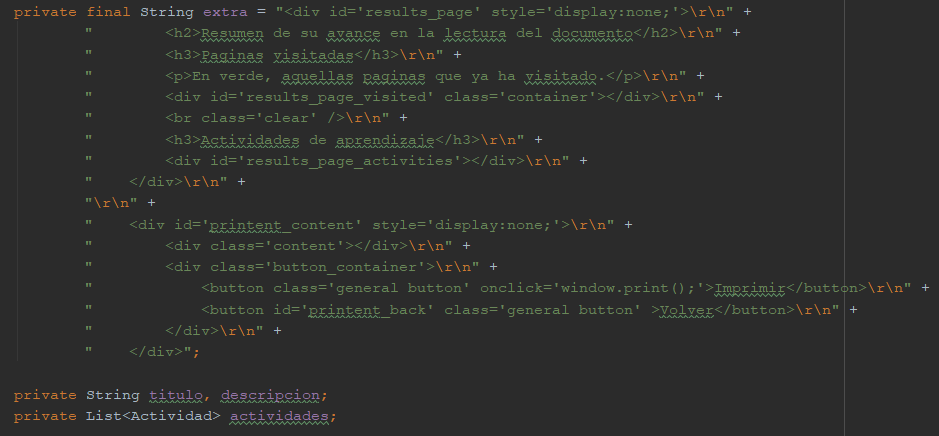






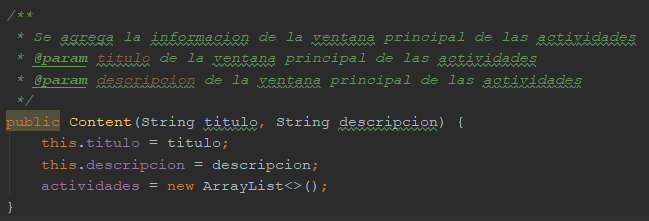






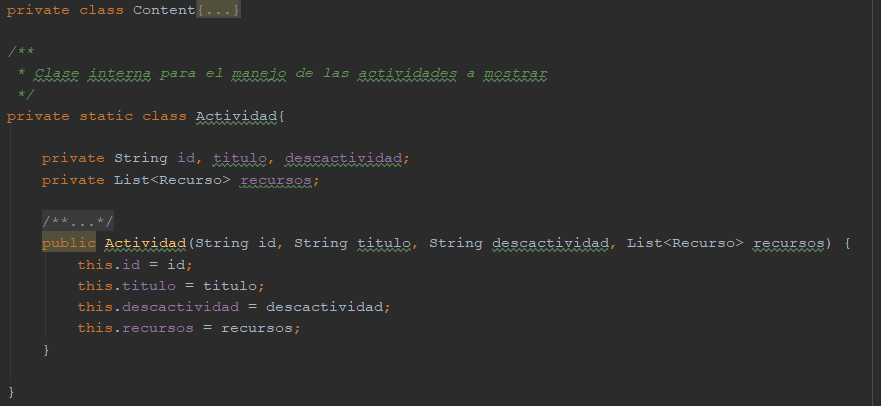
### El constructor

Y el constructor que guarda la información a mostrar en la sección principal de las actividades:



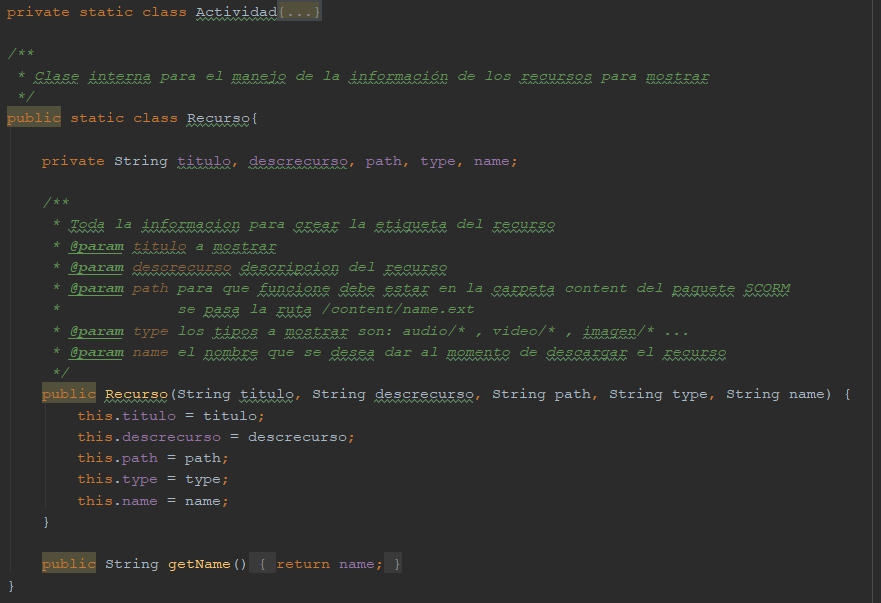
## Clase interna Actividad

Por otro lado, está la clase interna para las actividades que se van a mostrar con su constructor que guarda la información que se va a mostrar en la sección de la actividad correspondiente (cada instancia de *Actividad* es una sección para mostrar en la página content):



## Clase Recurso

Por último, la clase *Recurso* para el manejo de todos los recursos que tiene una actividad:



# Cibergrafía

<http://www.lingala.net/zip4j.html>

1. TIC: Tecnologías de la información y la comunicación [↑](#footnote-ref-1)
2. Se menciona acontinuación. [↑](#footnote-ref-2)