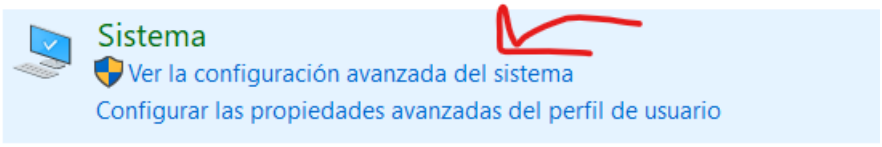
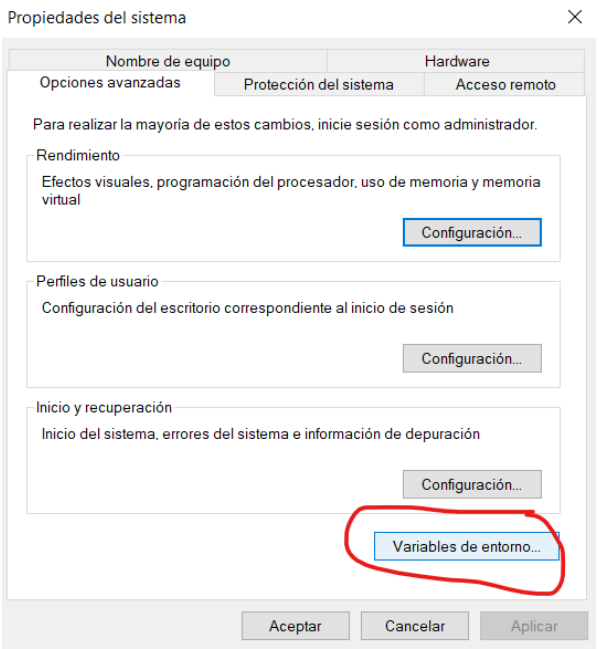
Ejercicio 2 PSP06

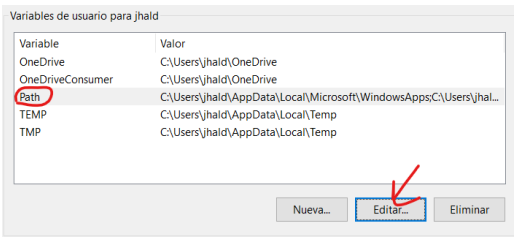
Antonio Jiménez Sevilla

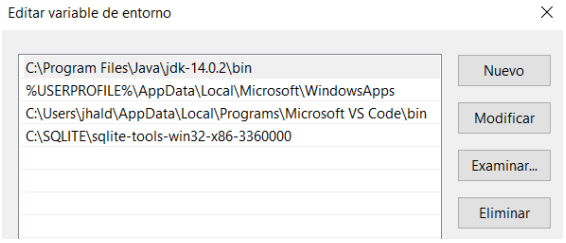
Configuramos la variable PATH para JAVA, vamos a panel de control/configuración avanzada del sistema/variables de entorno.



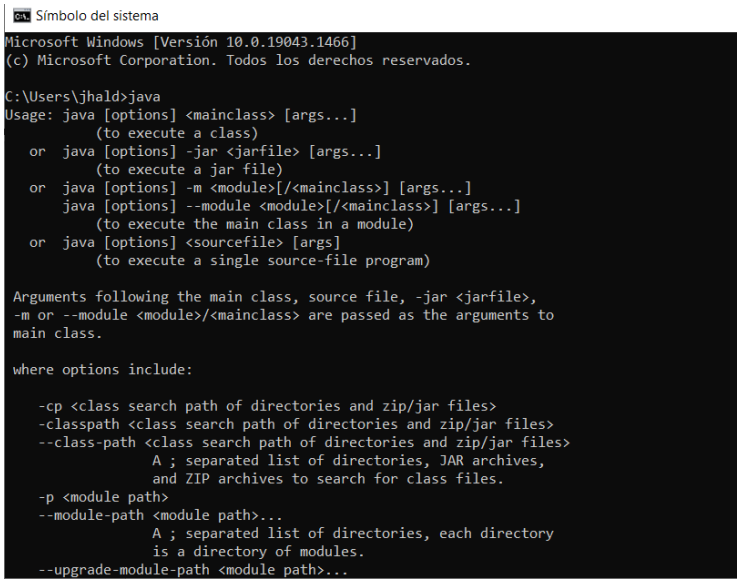


Editamos la variable PATH y añadimos la ruta del jdk 14.0.2.

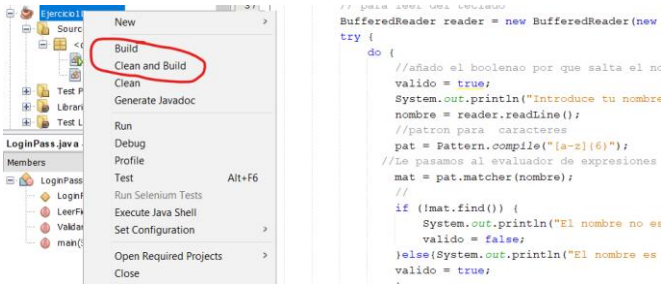




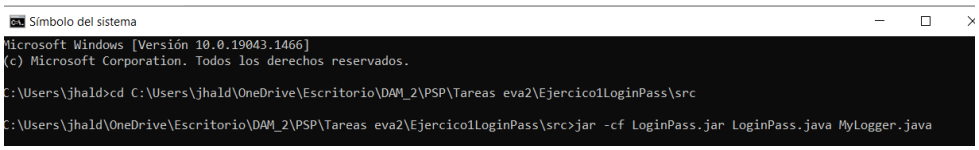
Comprobamos que funciona, escribiendo java en el CMD.

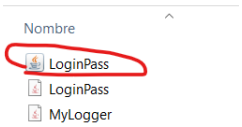


Ahora vamos a obtener el archivo.jar para realizar la firma digital en el CMD. Podemos hacerlo de dos maneras mediante un Build en el proyecto de netbeans o mediante la línea de comandos. Si lo hacemos en el proyecto de netbeans, nos creará una carpeta llamada dist donde encontraremos nnuestro archivo .jar.

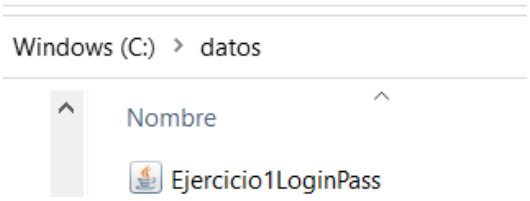


Si lo realizamos mediante el CMD de Windows se nos creará en la carpeta donde tenemos nuestras clases. Con el nombre que nosotros le hayamos dado. Pero nosotros vamos a utilizar el que hemos creado con el BUILD.





Una vez que tengamos nuestro archivo.jar,(el que hemos creado desde netbeans) voy a copiarlo y pegarlo en una carpeta en c: llamada datos.



Como podemos comprobar en nuestra carpeta c:/datos se encuentra nuestro archivo.jar.

```
c:\datos>dir
 El volumen de la unidad C es Windows
 El número de serie del volumen es: 24BE-3C7C

 Directorio de c:\datos

27/01/2022  11:51    <DIR>          .
27/01/2022  11:51    <DIR>          ..
20/01/2022  11:48             5.434 Ejercicio1LoginPass.jar
               1 archivos          5.434 bytes
               2 dirs  228.940.763.136 bytes libres
```

Para continuar con el ejercicio es importante que tengamos la misma versión de java, para compilar y para ejecutar.

```
C:\datos>java -version
java version "14.0.2" 2020-07-14
Java(TM) SE Runtime Environment (build 14.0.2+12-46)
Java HotSpot(TM) 64-Bit Server VM (build 14.0.2+12-46, mixed mode, sharing)

C:\datos>javac -version
javac 14.0.2
```

Para firmar nuestra aplicación utilizamos keytool que está dentro del directorio JDK. Escribiremos lo siguiente:

```
keytool -genkey -alias antFirmas -keypass pass1234 -keystore Firmas-PSP06 -storepass pass1234
```

-alias Indica el alias que se va a utilizar para referirnos al keystore, que es donde se van almacenar las llaves generadas.
-keypass indica la contraseña de la llave privada.
-keystore indica el keystore que ese esta creando.
-**storepass** indica el password del keystore.

Vemos que al introducir los datos nos da un error, eso es por que en las versiones de java superiores a 9 hay que indicarle el -keyalg rsa, esto quiere decir que el algoritmo para generar esa clave va ser un algoritmo rsa.

```
C:\datos>keytool -genkey -alias antFirmas -keypass pass1234 -keystore FirmasPSP06 -storepass pass1234
keytool error: java.lang.Exception: The -keyalg option must be specified.
```

Una vez introducido nos aparecen unas preguntas que tenemos que rellenar, una vez rellenados le decimos que si, en mi caso están en Inglés.

```
c:\datos>keytool -genkey -alias antFirmas -keypass pass1234 -keystore Firmas-PSP06 -storepass pass1234 -keyalg rsa
What is your first and last name?
  [Unknown]:  AntJimenez
What is the name of your organizational unit?
  [Unknown]:  Departamento de Informatica
What is the name of your organization?
  [Unknown]:  I.E.S Augusto G Linares
What is the name of your City or Locality?
  [Unknown]:  Santander
What is the name of your State or Province?
  [Unknown]:  Cantabria
What is the two-letter country code for this unit?
  [Unknown]:  ES
Is CN=AntJimenez, OU=Departamento de Informatica, O=I.E.S Augusto G Linares, L=Santander, ST=Cantabria, C=ES correct?
  [no]:  yes

Generating 2.048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 90 days
        for: CN=AntJimenez, OU=Departamento de Informatica, O=I.E.S Augusto G Linares, L=Santander, ST=Cantabria, C=ES
```

Podemos comprobar que tenemos ya nuestro fichero .jar y nuestro archivo de firmas.

```
c:\datos>dir
 El volumen de la unidad C es Windows
 El número de serie del volumen es: 24BE-3C7C

 Directorio de c:\datos

27/01/2022  11:54    <DIR>          .
27/01/2022  11:54    <DIR>          ..
20/01/2022  11:48             5.434 Ejercicio1LoginPass.jar
27/01/2022  11:54             2.677 Firmas-PSP06
               2 archivos          8.111 bytes
               2 dirs  228.939.251.712 bytes libres

c:\datos>
```

Ahora vamos realizar la firma del fichero con el siguiente comando.

```
jarsigner -keystore Firmas-PSP06 -signedjar sFirmasPSP06.jar Ejercicio1LoginPass.jar antFirmas
```

Escribimos el keystore que es el repositorio de firmas que hemos creado anteriormente.

-signedjar es el nombre que le vamos a dar a nuestro fichero jar firmado.

Después poner el fichero que queremos firmar. Y ponemos el nombre del alias que hemos escrito al principio, para indicar con que nombre o que alias vamos a firmar.

```
c:\datos>jarsigner -keystore Firmas-PSP06 -signedjar sFirmasPSP06.jar Ejercicio1LoginPass.jar antFirmas
Enter Passphrase for keystore:
jar signed.

Warning:
The signer's certificate is self-signed.

c:\datos>
```

Como podemos comprobar, nos ha pedido la contraseña que creamos anterior mente y se puede observar como se ha firmado correctamente el fichero.

Hago un dir para comprobar que se ha creado correctamente todo. Tengo el fichero .jar original, el firmado sFirmasPSP06 y mi repositorio de firmas.

```
c:\datos>dir
 El volumen de la unidad C es Windows
 El número de serie del volumen es: 24BE-3C7C

 Directorio de c:\datos

27/01/2022  12:00    <DIR>          .
27/01/2022  12:00    <DIR>          ..
20/01/2022  11:48             5.434 Ejercicio1LoginPass.jar
27/01/2022  11:54             2.677 Firmas-PSP06
27/01/2022  12:00             7.232 sFirmasPSP06.jar
               3 archivos         15.343 bytes
               2 dirs  228.930.506.752 bytes libres

c:\datos>
```

Para poder mandar esto, a una persona que quiera ejecutar este fichero firmado por mi sería la clave publica de ese fichero generado por mi para que pueda validar mi firma. Para exportarlo se usa el siguiente código.

```
keytool –export –keystore Firmas-PSP06 –alias antFirmas –file Anto-
nio.cert
```

```
c:\datos>keytool -export -keystore Firmas-PSP06 -alias antFirmas -file Antonio.cert
Enter keystore password:
Certificate stored in file <Antonio.cert>
```

Al hacer dir vemos que se ha generado correctamente mi certificado de firma.

```
c:\datos>dir
 El volumen de la unidad C es Windows
 El número de serie del volumen es: 24BE-3C7C

 Directorio de c:\datos

27/01/2022  12:02    <DIR>          .
27/01/2022  12:02    <DIR>          ..
27/01/2022  12:02               974 Antonio.cert
20/01/2022  11:48             5.434 Ejercicio1LoginPass.jar
27/01/2022  11:54             2.677 Firmas-PSP06
27/01/2022  12:00             7.232 sFirmasPSP06.jar
               4 archivos         16.317 bytes
               2 dirs  228.936.024.064 bytes libres

c:\datos>
```

Ahora para el siguiente punto, que solo pueda leer los datos de c:/datos Una vez que tenemos el fichero jar firmado (sFirmasPSP06.jar) y el certificado de seguridad (Antonio.cert), podemos intentar ejecutar directamente la aplicación, pero veremos que al utilizar el SecurityManger nos da problemas de seguridad.

Como lo hemos realizado, en jdk14 deberemos de introducir los datos de Policytool a mano. Pero primero he creado una carpeta que se llama importar donde he introducido el fichero .jar firmado, y el keystore.

Para que la aplicación funcione correctamente, primero debemos importar el certificado ejecutando:

```
keytyool -import -alias AntJimenez -file Antonio.cert

-keystore Firmas-PSP06
```

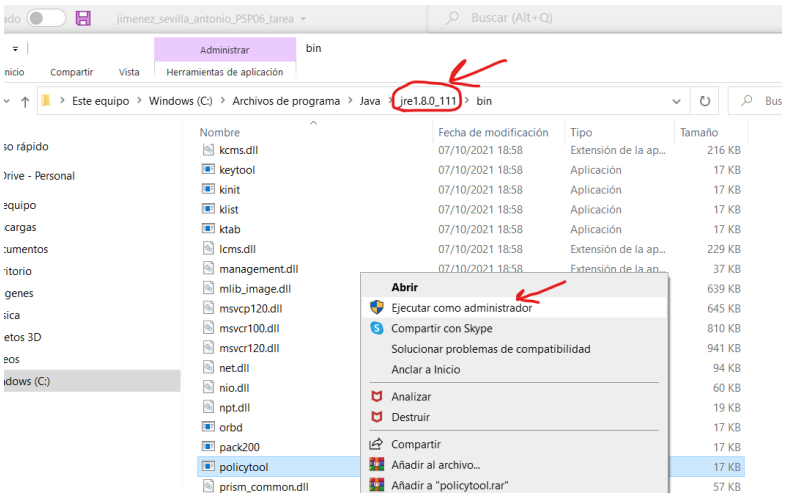
```
c:\datos>cd c:\datos\importar

c:\datos\importar>keytool -import -alias AntJimenez -file Antonio.cert -keystore Firmas-PSP06
Enter keystore password:
Re-enter new password:
Owner: CN=AntJimenez, OU=Departamento de Informatica, O=I.E.S Augusto G Linares, L=Santander, ST=Cantabria, C=ES
Issuer: CN=AntJimenez, OU=Departamento de Informatica, O=I.E.S Augusto G Linares, L=Santander, ST=Cantabria, C=ES
Serial number: dfa06e3b11d6837d
Valid from: Thu Jan 27 11:54:33 CET 2022 until: Wed Apr 27 12:54:33 CEST 2022
Certificate fingerprints:
         SHA1: B3:CC:BC:A4:47:BE:83:E5:8C:74:B4:88:8D:FB:30:A9:80:1E:C0:87
         SHA256: 02:C3:98:99:D2:0E:78:BC:1D:44:2B:C8:0C:82:CE:94:51:28:47:A7:DC:D9:A1:F3:D6:E2:33:4A:52:64:D0:35
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: D7 03 76 55 F5 02 D2 DF   47 A0 94 51 EB 4E 7C C7  ..vU....G..Q.N..
0010: F3 36 80 12                                        .6..
]
]

Trust this certificate? [no]:  yes
```

```
c:\datos\importar>dir
 El volumen de la unidad C es Windows
 El número de serie del volumen es: 24BE-3C7C

 Directorio de c:\datos\importar

27/01/2022  12:30    <DIR>          .
27/01/2022  12:30    <DIR>          ..
27/01/2022  12:02               974 Antonio.cert
27/01/2022  12:30             1.274 Firmas-PSP06
27/01/2022  12:00             7.232 sFirmasPSP06.jar
               3 archivos          9.480 bytes
               2 dirs  228.929.630.208 bytes libres

c:\datos\importar>_
```

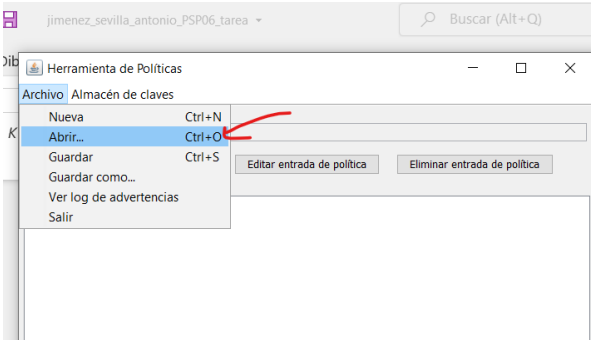
Para agregar los permisos de lectura de la carpeta c:/datos, lo que hacemos, es crear un archivo policy con el siguiente código. Donde lo agregaremos a mas adelante a la carpeta del proyecto.

Para realizar el siguiente paso, voy a usar el sistema de netbeans jdk 1.8 para realizar la política de seguridad.

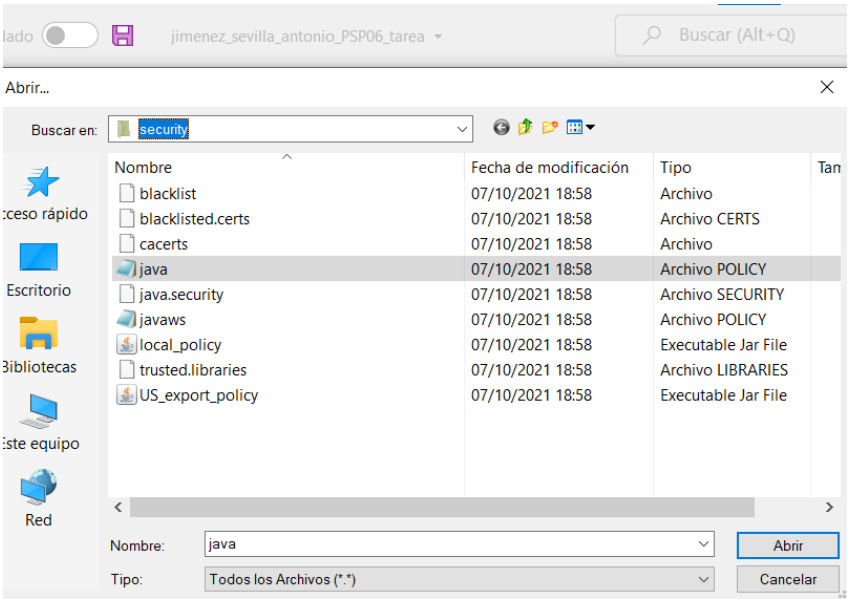
Para ello vamos a la carpeta de jre 1.8, ya que es el java que se está ejecutando. en mi caso está en la carpeta bin, el archivo policytool, el cual con el botón derecho clicamos y lo ejecutamos como administrador.



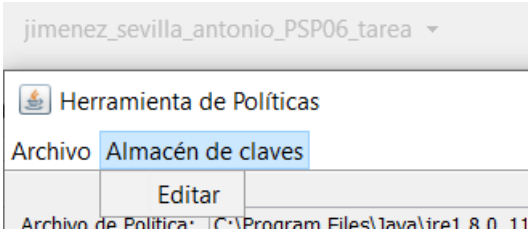
Una vez abierto nos aparecerá la siguiente pantalla, donde tendremos que abrir el archivo de políticas de seguridad.



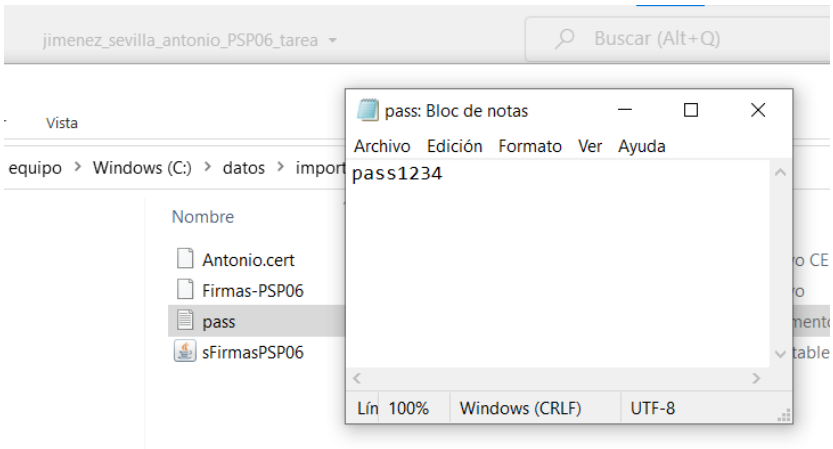
Este archivo se encuentra en la carpeta security del jre.1.8.



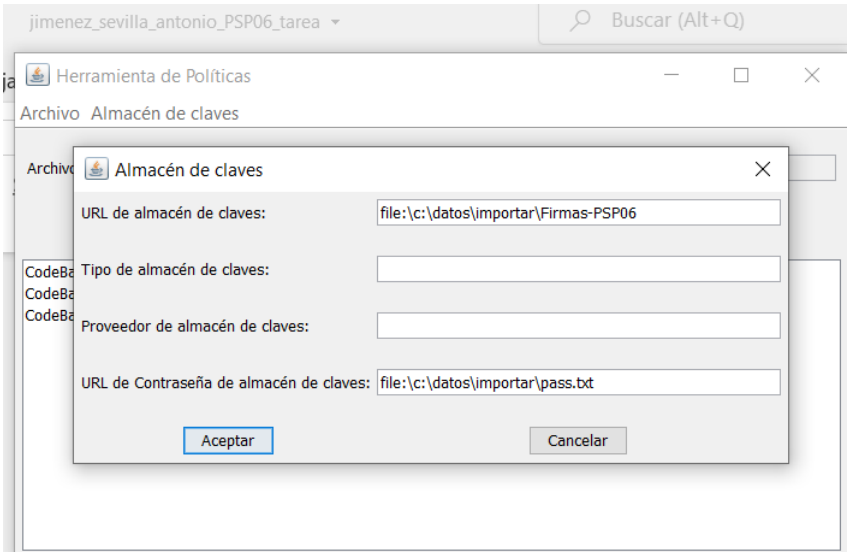
Le damos abrir y una vez abierto como vamos a añadir permisos para determinadas firmas, tenemos que configurar como se accede al almacén de claves, que hemos creado con anterioridad.



Una vez seleccionamos editar, escribimos la url de nuestro almacén de claves creado con anterioridad, que se llama Firmas-PSP06. Hay que crear un archivo para guardar la contraseña, para que pueda leer la clave publica y pueda leer un archivo firmado. Yo he creado un archivo llamado pass.txt con la contraseña pass1234. Indicamos la url de este archivo también.



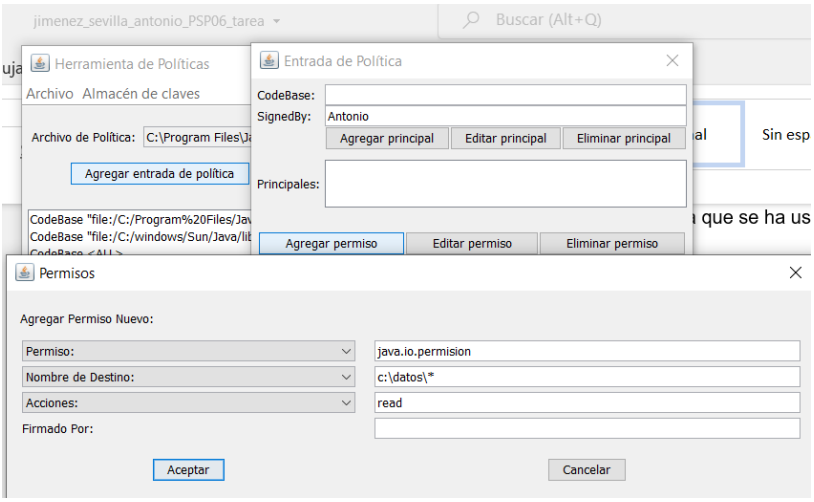
El tipo de almacen de claves es lo que mas tarde cambiaremos, ya que java trae uno por defecto y al tener que hacerlo mano más adelante, tendremos que cambiar el tipo de clave a pkcs12. Debido a que se ha usado un jdk posterior a 10.



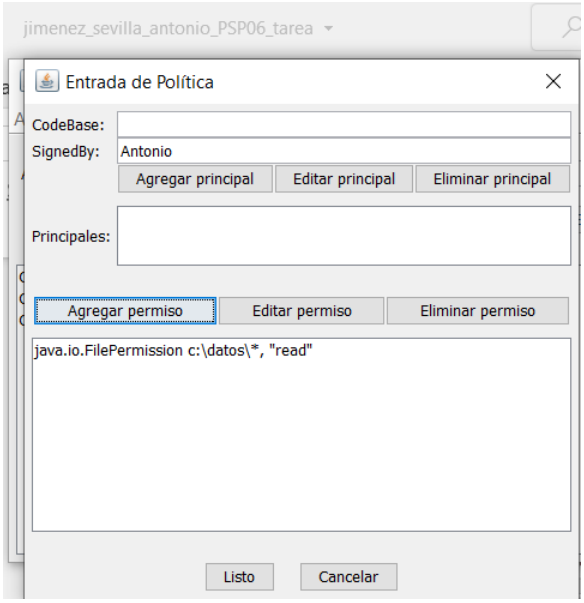
Si todo es correcto al darle aceptar no nos da ningún problema.

Ahora tenemos que agregar la política de seguridad, para permitir que cualquier fichero firmado por la clave pública que hemos importado, pueda acceder al directorio. Clicamos en agregar política, se nos abrirá una ventana, en la que escribiremos en la casilla signedBy, el nombre del alias que le hemos puesto a la clave pública Antonio.

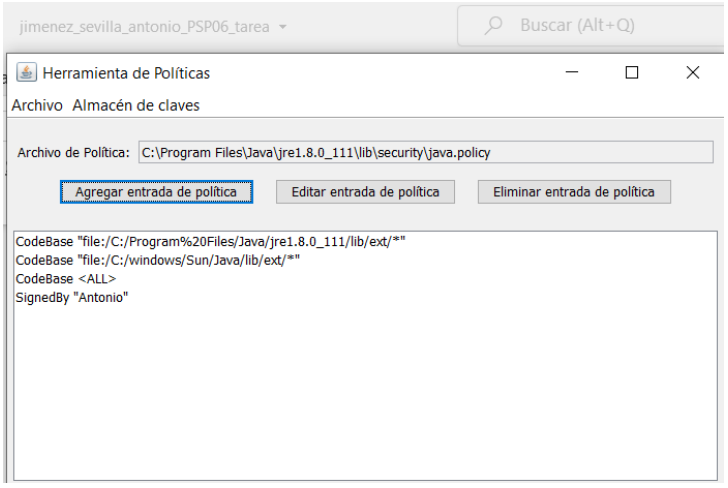
Le agregamos un permiso, el destino, ponemos un asterisco al final para aplicarlo a todos los archivos y carpetas del directorio. Y para las acciones escribimos read. Le damos aceptar.



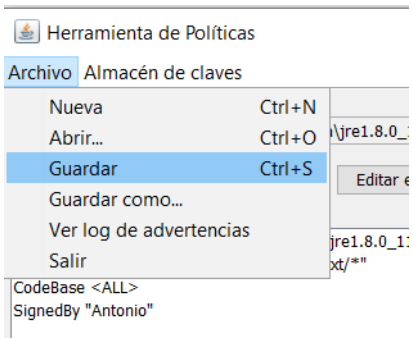
Le damos a listo.

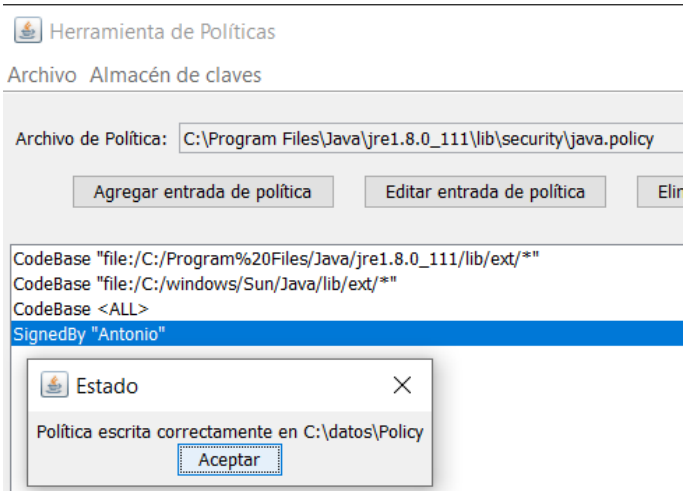


Y nos creado una política nueva que cualquier proyecto firmado por Antonio podrá leer en la carpeta datos.

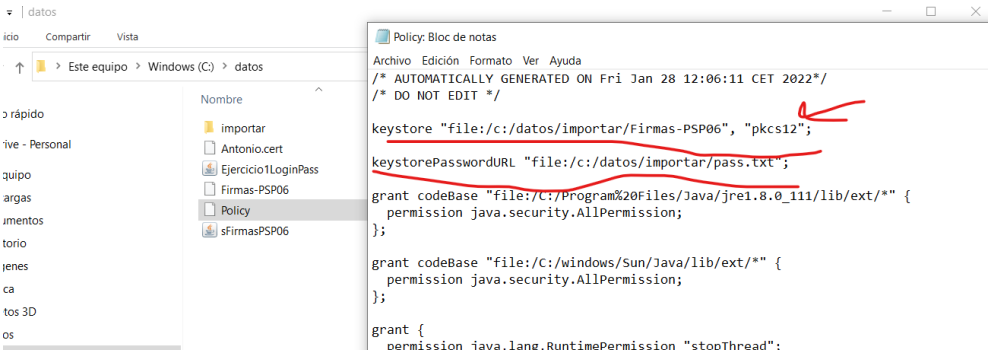


Le damos a guardar como y lo guardamos en nuestro directorio datos.





Ahora vamos al archivo Policy guardado y cambiamos las líneas donde hemos creado nuestra política. Ya que al usar versiones posteriores a 10 hay que poner pkcs12 en vez de la que viene por defecto con Java.




```
grant signedBy "Antonio" {
  permission java.io.FilePermission "c:\\datos\\*", "read";
};
```