

Consumo local libs RSI

Como consumir librerías privadas (libs propios) desde proyectos backend RSI

Consumo local de libreria privada

El presente documento tiene como objetivo guiar a cada uno de los desarrolladores del equipo de RSI en la configuración del entorno de trabajo, de manera que puedan realizar los ajustes pertinentes para lograr un correcto acoplamiento de la librería.

1. Generación del token de GitLab.

En primer lugar es necesario crear un token, para ello el usuario debe ingresar a GitLab, tras iniciar sesión, debe presionar en la imagen de perfil y allí debe ingresar en la sección de “Preferencias”.

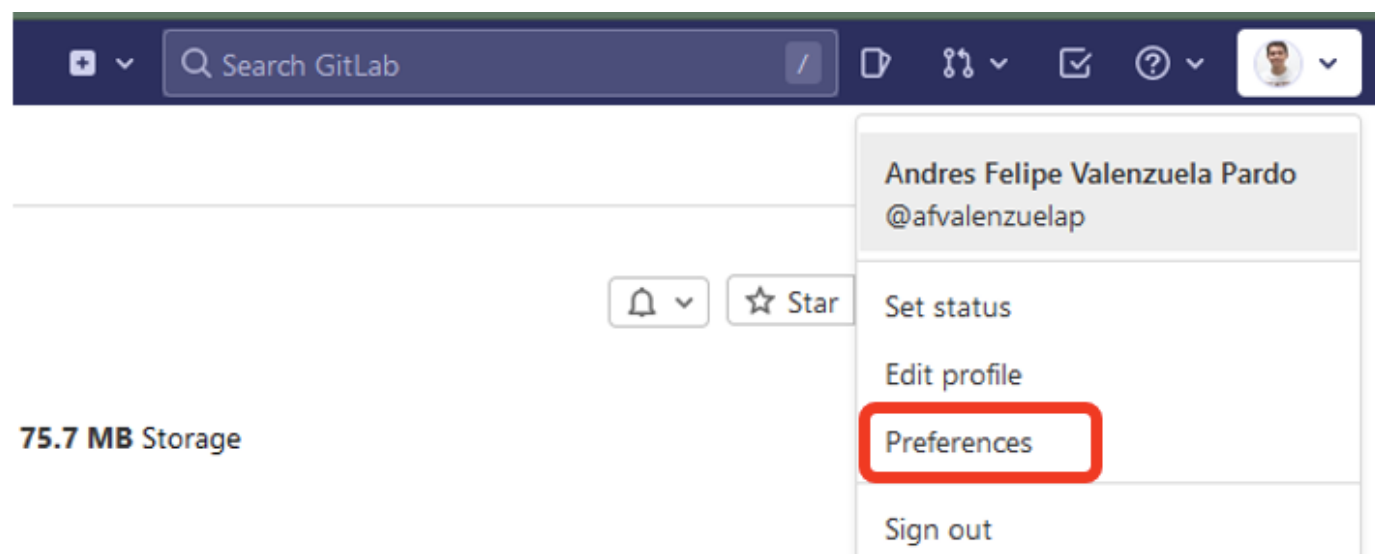
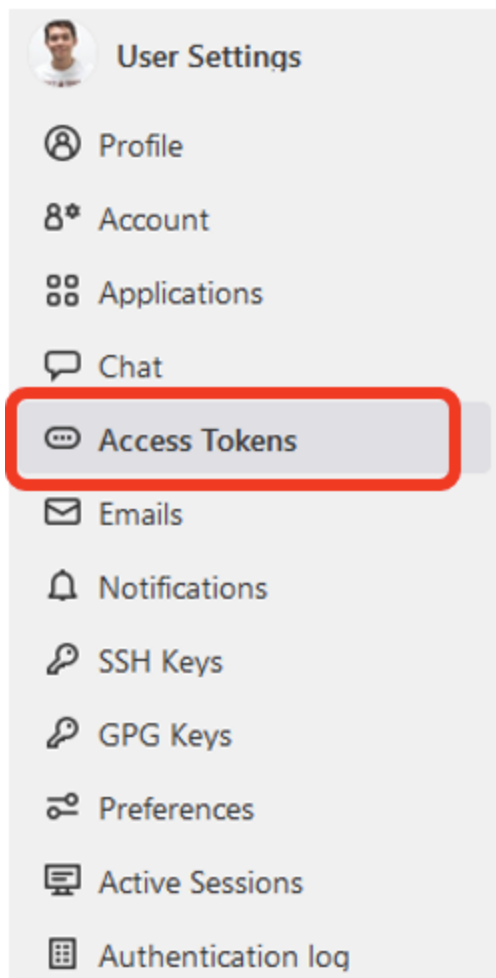


Imagen 1. Esta sección se encuentra en la esquina superior derecha de la página.

Al ingresar en el apartado de preferencias, aparecerá un menú en el lateral izquierdo, en este se debe dar click en la opción “Access Tokens”.



User Settings > Access Tokens

Personal Access Tokens

You can generate a personal access token for each application you use that needs access to the GitLab API.

You can also use personal access tokens to authenticate against Git over HTTP. They are only accepted when you have Two-Factor Authentication (2FA) enabled.

Imagen 2. Menú de configuraciones de usuario ubicado en el lateral izquierdo

En la sección de Tokens se procederá a generar un nuevo token, para ello se le debe asignar un nombre y se debe señalar que dicho token será de tipo “read api”, tal y como se ve en la imagen 3, seguidamente se debe dar click en el botón “Create personal Access token”

Personal Access Tokens

You can generate a personal access token for each application you use that needs access to the GitLab API.

You can also use personal access tokens to authenticate against Git over HTTP. They are the only accepted password when you have Two-Factor Authentication (2FA) enabled.

Add a personal access token

Enter the name of your application, and we'll return a unique personal access token.

Token name

For example, the application using the token or the purpose of the token. Do not give sensitive information for the name of the token, as it will be visible to all project members.

Expiration date

Select scopes

Scopes set the permission levels granted to the token. [Learn more.](#)

☐ api

Grants complete read/write access to the API, including all groups and projects, the container registry, and the package registry.

☒ read_api

Grants read access to the API, including all groups and projects, the container registry, and the package registry.

☐ read_user

Grants read-only access to the authenticated user's profile through the /user API endpoint, which includes username, public email, and full name. Also grants access to read-only API endpoints under /users.

☐ read_repository

Grants read-only access to repositories on private projects using Git-over-HTTP or the Repository Files API.

☐ write_repository

Grants read-write access to repositories on private projects using Git-over-HTTP (not using the API).

☐ read_registry

Grants read-only access to container registry images on private projects.

☐ write_registry

Grants write access to container registry images on private projects.

Imagen 3. Sección de tokens personales de acceso.

Posteriormente, la página se recargará y mostrará en la parte superior el token generado.

“

Es importante que se copie el token, ya que este solo será visible esta vez, en caso de perder el token generado, el usuario deberá crear nuevamente otro token.

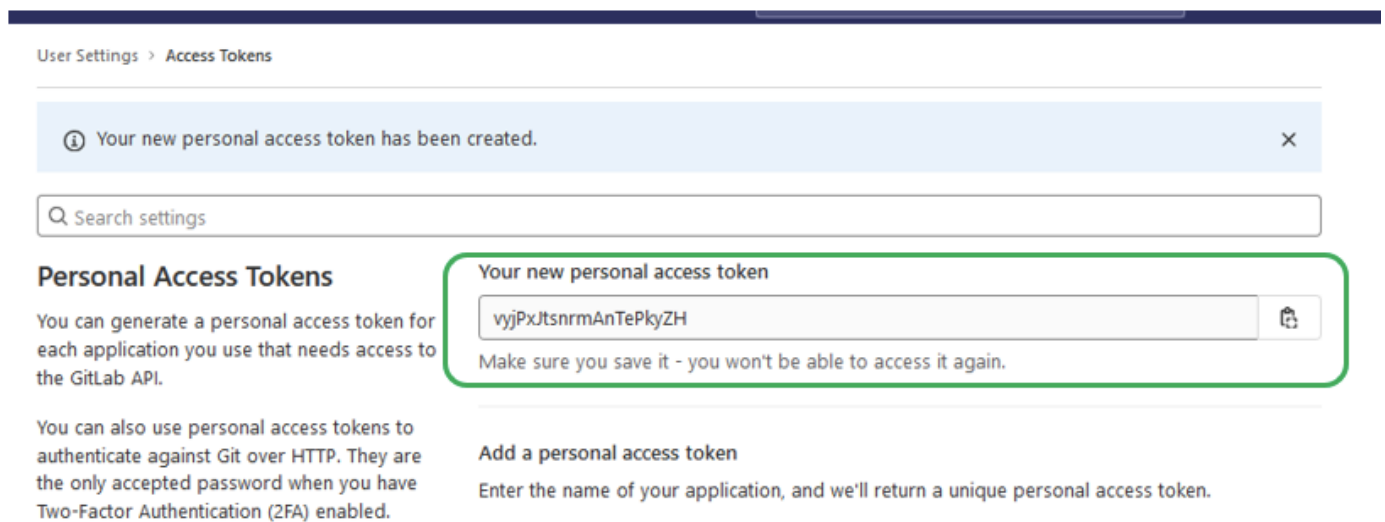


Imagen 4. Token de acceso personal generado.



El token mostrado en la imagen es un ejemplo (no viable) y debe ser creado por cada desarrollador BACK para el consumo de la librería.

2. Ubicar y ajustar archivo `settings.xml` en el directorio `“.m2”`.

Debemos ubicar el archivo `settings.xml` en el directorio `.m2`, de manera que Maven pueda reconocer las librerías privadas correctamente. Para ello luego de ubicarlo reemplazamos su contenido como se indica

Windows

En el caso de Windows, el proceso para ubicar la carpeta `.m2` es el siguiente:

Primero se debe ubicar en el disco duro donde se encuentre el sistema operativo (generalmente el disco C), en este se debe entrar en la capeta “Usuarios” tal y como se ve en la dirección de la imagen 5.

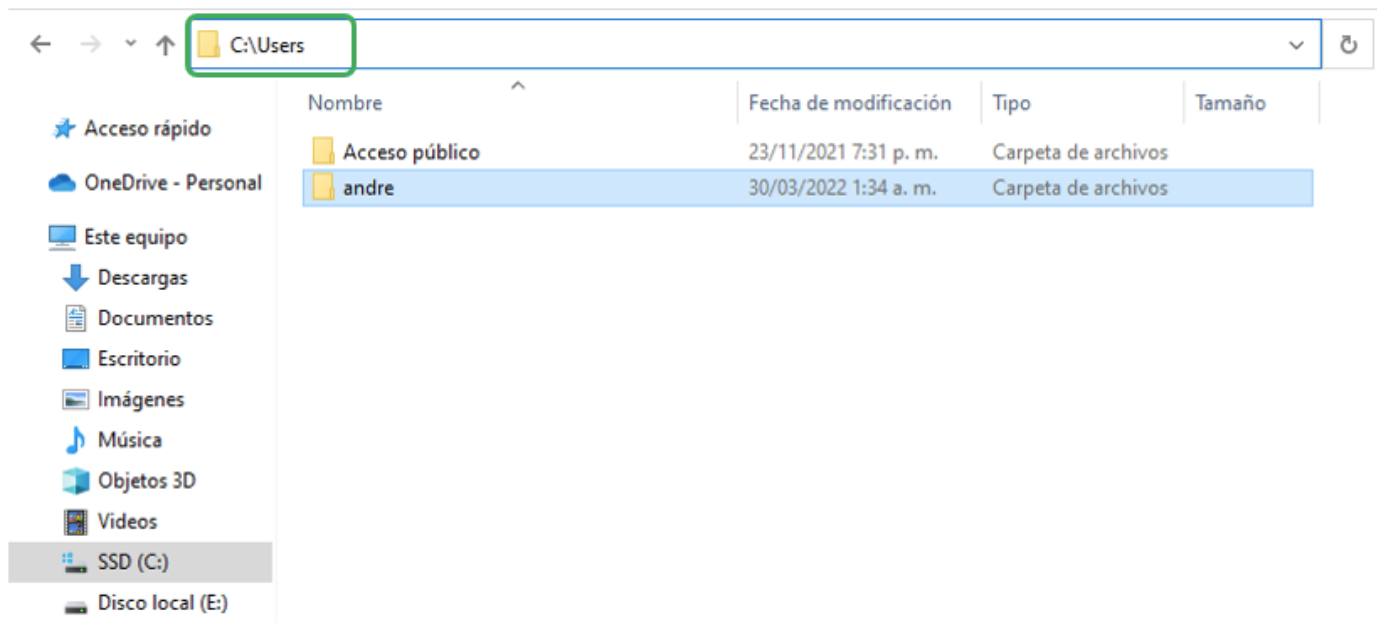


Imagen 5. Carpeta de usuarios.

Se debe ingresar en la carpeta de usuario correspondiente a cada caso (para este caso en particular se llama “andre”), una vez ingresado en dicha carpeta, se podrá localizar la carpeta .m2.

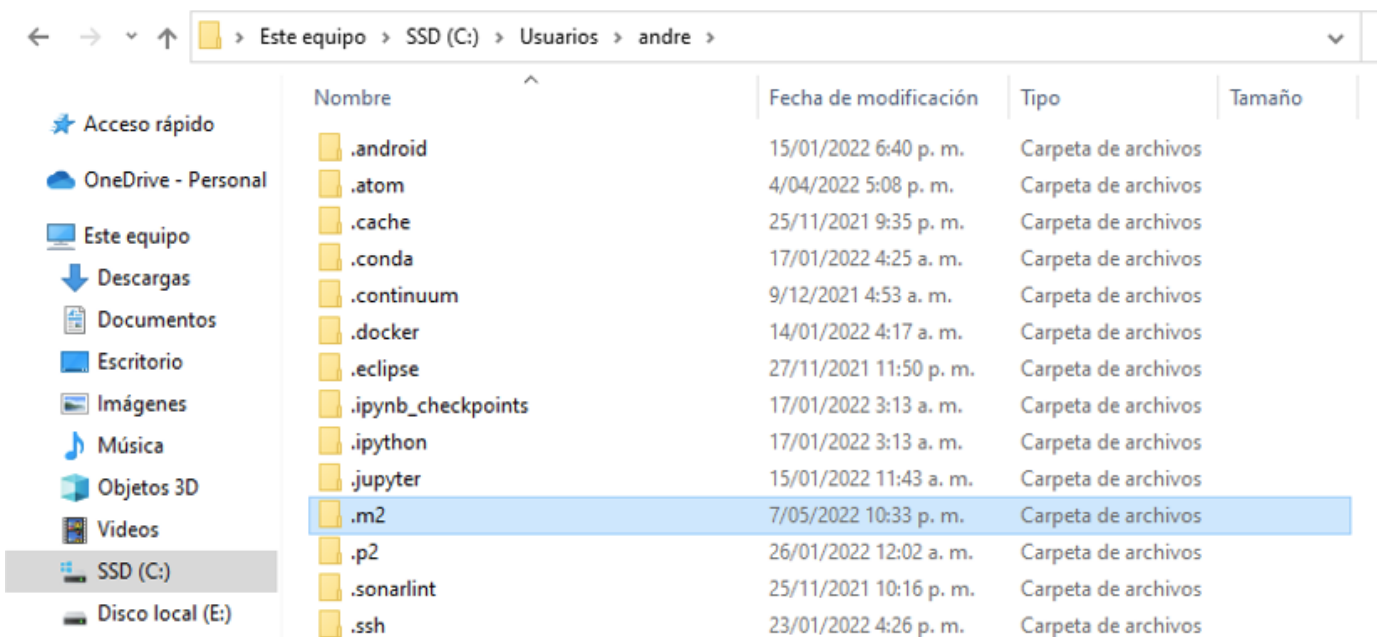


Imagen 6. Carpeta de archivos de usuario.

Una vez dentro de la carpeta .m2, se pegará el archivo *settings.xml* modificado con el token personal ó se reemplaza su contenido como se muestra más adelante.

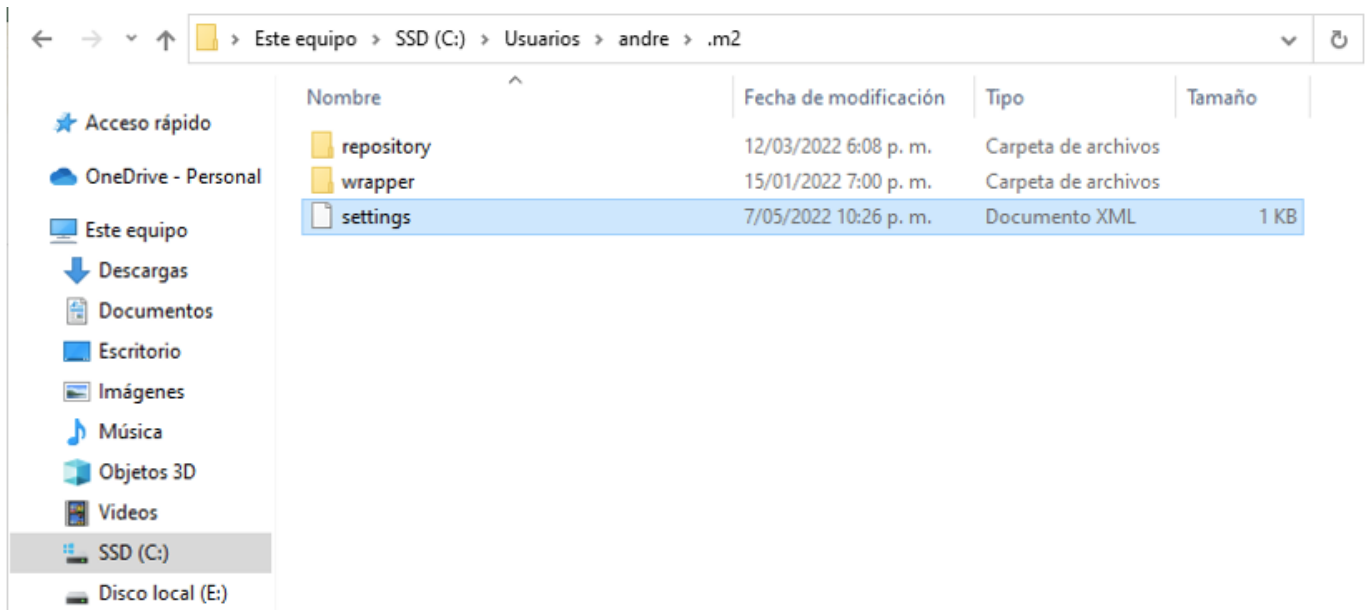


Imagen 7. Carpeta .m2 con el *settings.xml* insertado.

Linux

El procedimiento es similar en este sistema operativo, para empezar, se debe ir al directorio principal el cual suele estar referenciado como `"/home/{username}/"`, para la imagen de ejemplo la ruta del directorio principal es: `"/home/dell/"`.

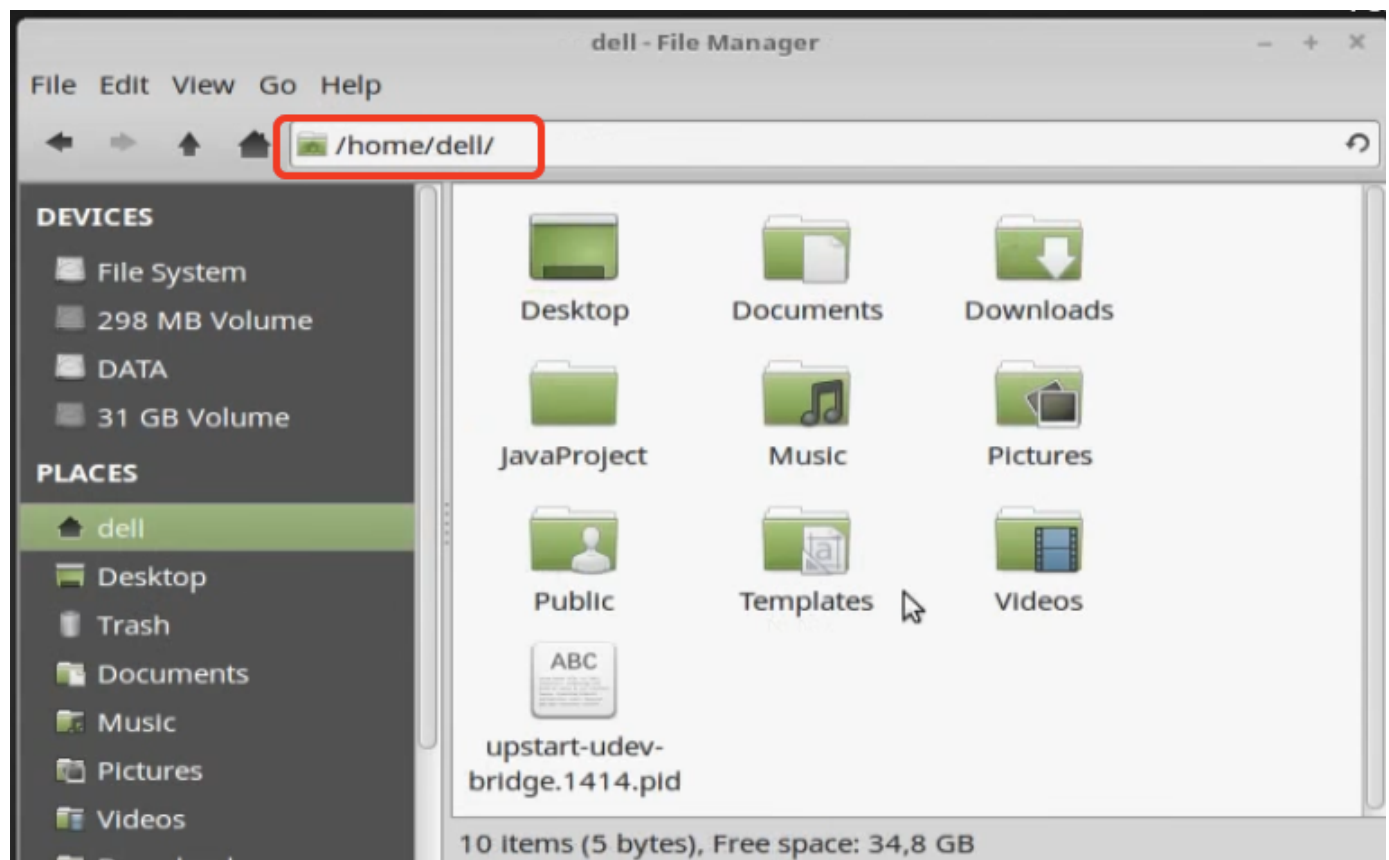


Imagen 8. Vista del directorio principal en Linux Mint.

La carpeta .m2 suele estar oculta junto con otros directorios, por esto, **si en principio no aparece**, entre a la opción de “view” y habilite la opción “show hidden files”.

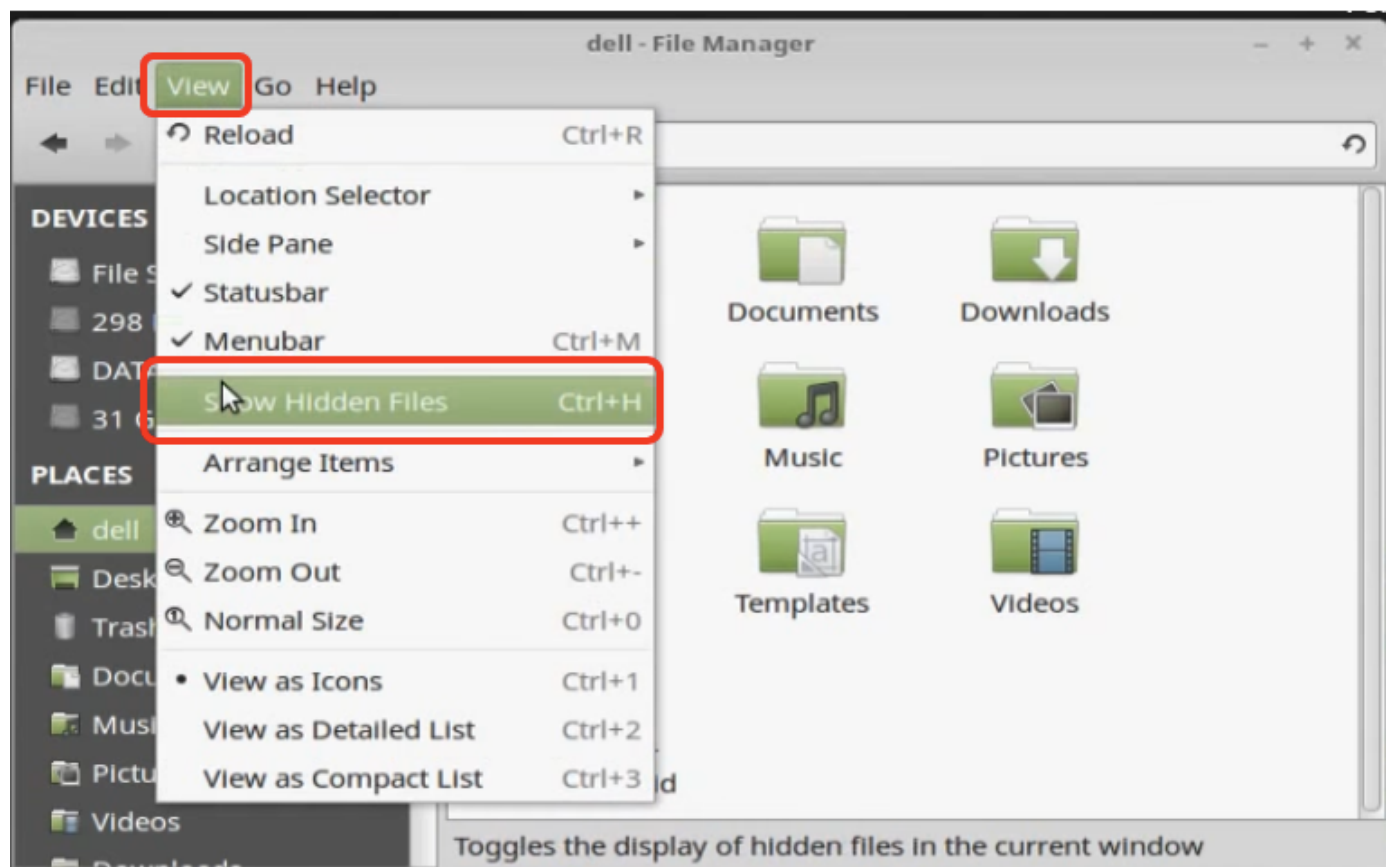


Imagen 9. Metodo para mostrar directorios ocultos.

Una vez habilitada la vista de archivos ocultos, se podrá identificar el directorio .m2, tal y como se observa en la imagen 10.

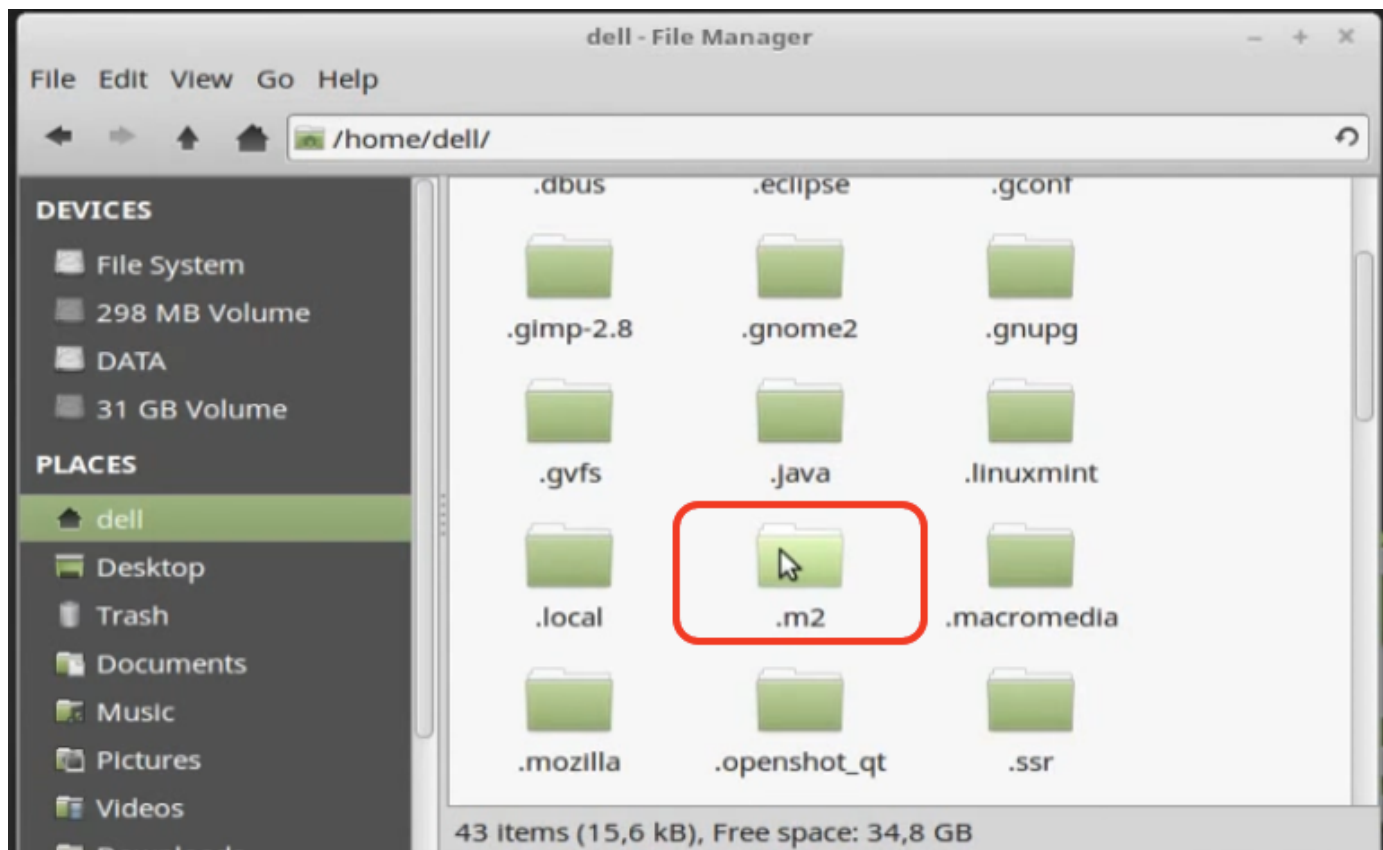


Imagen 10. Selección del directorio `.m2`.

Finalmente se vuelve a insertar el archivo `settings.xml` en el directorio `.m2` tal (modificado) ó reemplazamos su contenido como se hizo en Windows.

3. Configuración del archivo “`settings.xml`”.

Una vez sea ubicado correctamente el archivo `settings.xml` es necesario configurarlo, al abrir el archivo, este debe contener en su contenido algo similar a lo siguiente:

```
1  <settings>
2      <servers>
3          <server>
4              <id>gitlab-maven</id>
5              <configuration>
6                  <httpHeaders>
7                      <property>
8                          <name>Private-Token</name>
9                          <value>EL_TOKEN_DE_GIT_PARA_USUARIO</value>
10                     </property>
11                 </httpHeaders>
12             </configuration>
13         </server>
```



```
14     </servers>
15 </settings>
```

En este se puede observar que entre las etiquetas “value” se encuentra un texto el cual advierte la necesidad de un token de Git, es allí donde se pondrá el token de GitLab anteriormente generado. De manera que el archivo quedaría de la siguiente manera:

```
1 <settings>
2   <servers>
3     <server>
4       <id>gitlab-maven</id>
5       <configuration>
6         <httpHeaders>
7           <property>
8             <name>Private-Token</name>
9             <value>xxyyyyyzzzEJEMPL0ooo</value>
10          </property>
11        </httpHeaders>
12      </configuration>
13    </server>
14  </servers>
15 </settings>
```

Por ultimo se guarda el archivo para que el token quede debidamente almacenado.



Sobra aclarar que el token almacenado es el generado en la primera sección por cada uno

4. Actualización de dependencias.

Una vez realizados los cambios anteriores, se procede a recargar la configuración de Maven para que detecte los cambios realizados.

Para ello el procedimiento es: click-derecho en el *pom.xml* → *Maven* → *Reload project* o *Update project*.

IntelliJ

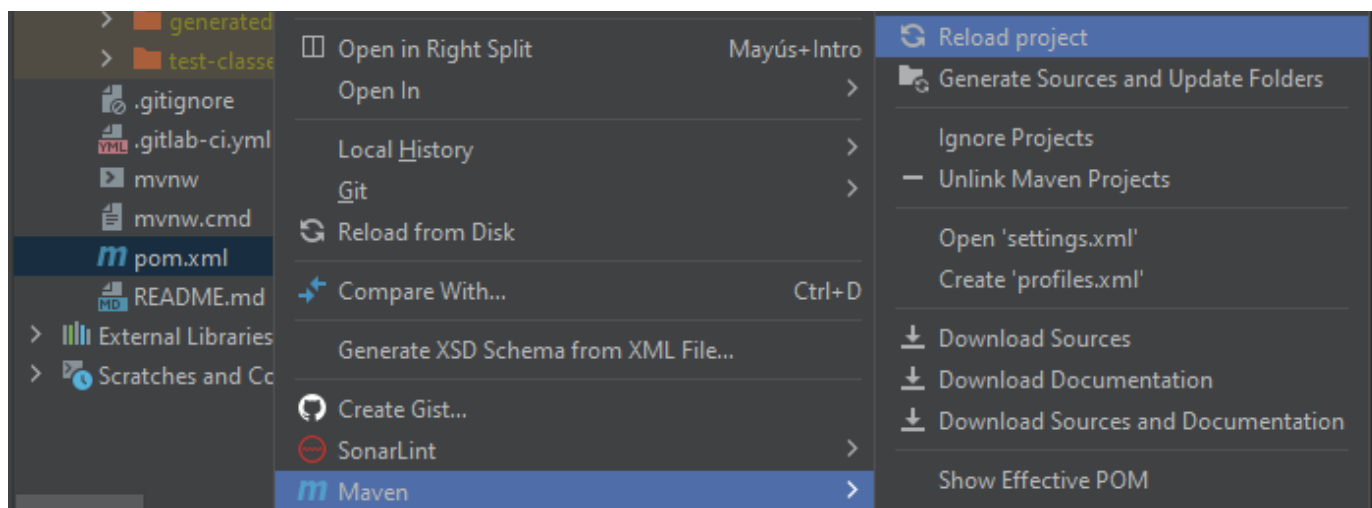


Imagen 11. Metodo para recargar dependencias del proyecto en IntelliJ.

Eclipse

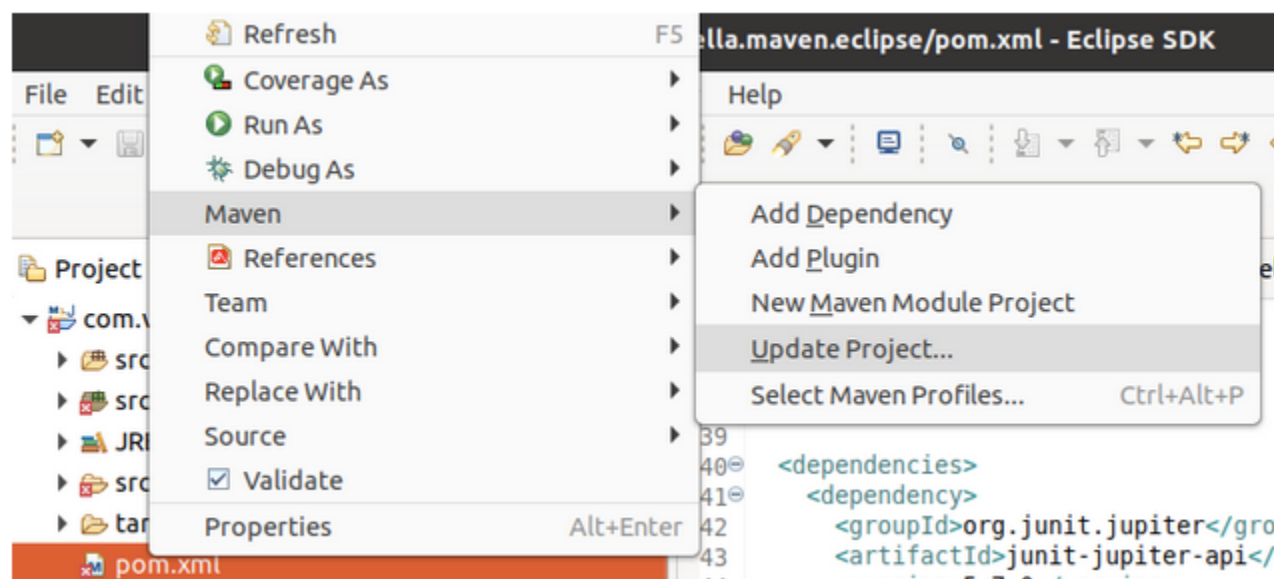


Imagen 12. Metodo para recargar dependencias del proyecto en Eclipse.



Una vez realizados cada uno de los anteriores pasos, el proyecto estará listo para ejecutarse de manera correcta. De igual forma no olvidar estar haciendo mvn clean y mvn install cada vez que se requiera.

5. Versionamiento Semántico.

El versionamiento semántico es un estándar hecho para definir la versión de una biblioteca. Se compone de 3 cifras de números enteros positivos separados por puntos, estos números se incrementan en 1 cada vez que una biblioteca sufre cambios y se publican.

Dependiendo de la naturaleza del cambio que se haya introducido, se pueden identificar 3 tipos de cambios:

ej:

1.0.2

MAJOR . MINOR . PATCH

- **Patch:** Hace referencia a la corrección de errores o *bugs*, **estos cambios son compatibles** con la versión anterior.

```
1 | 1.3.1 // Versión actual con errores
2 | 1.3.2 // Versión nueva con correcciones
```

- **Minor:** Cambio que añade alguna característica nueva a la biblioteca o modifica alguna ya existente, Aunque se aumente esta versión todos **los cambios son compatibles** con las anteriores. Si se aumenta la versión “*minor*” se debe reiniciar el “*patch*”.

```
1 | 1.2.7 // Versión anterior
2 | 1.3.0 // Versión nueva compatible con versiones anteriores, se reinicia el patch
```

- **Major:** Cambio drástico en la biblioteca. Este cambio **puede ocasionar errores** para los desarrolladores que utilicen versiones anteriores a esta. Cuando se incrementa la versión “*major*”, entonces “*minor*” y “*patch*” se reinician.

```
1 | 1.5.4 // Versión anterior
2 | 2.0.0 // Versión nueva, se reinicia el minor y el patch
```

Identificadores de estabilidad

Además de poder definir los cambios en el código mediante *major*, *minor* o *patch*, se suelen añadir unos identificadores que ayudan a marcar versiones específicas, **indicando la estabilidad de dicha versión**. Para utilizarlos se agrega un guion y una palabra (indicador) luego del *patch*.

- **Alpha:** Es una **versión inestable** que es muy probable que tenga muchas opciones que mejorar.

```
1 | 1.0.0-alpha
```

- **Beta:** Es una **versión mas estable** que *Alpha*, en esta se realizan pruebas de rendimiento, usabilidad y

funcionamiento de algunos módulos.

1 | 1.0.0-beta

- **Pre-release:** Es el último paso de la librería antes de salir a producción. Su identificador es “rc” (*Release Candidate*),

1 | 1.0.0-rc

- **snapshot:** Indica que se están haciendo cambios en la librería (pruebas)

1 | 1.0.0-snapshot

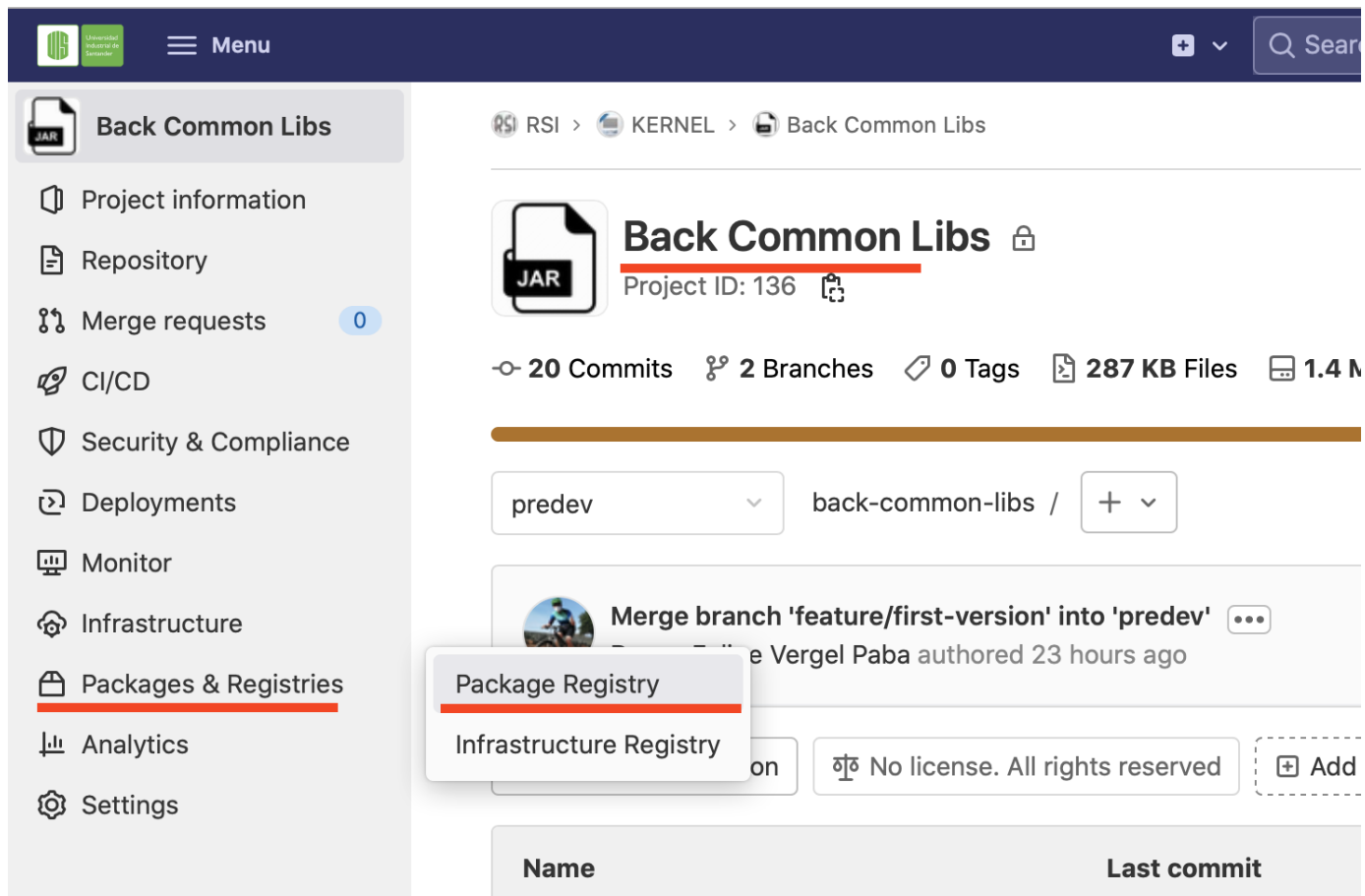
- **stable:** Es la versión que consumen o pronto consumirán los proyectos backend. No tiene la palabra reservada en los proyectos RSI:

1 | 1.0.0



En los proyectos RSI por simplicidad, las librerías privadas RSI solo manejaremos snapshots y stables. En todo caso cuando haya lugar de migrar a una versión superior de la misma, lo harán determinadas personas o líderes de desarrollo, siendo transparente para todos el cambio (sólo hacer update en maven), pero si es importante conocer la nomenclatura para conocer de las mismas.


Si se quiere visualizar las distintas librerías generadas podemos visualizarlas como muestra la imagen en el proyecto BACK COMMON LIBS (siempre y cuando tengamos acceso)



Back Common Libs


Project information
Repository
Merge requests 0
CI/CD
Security & Compliance
Deployments
Monitor
Infrastructure
Package Registry
Analytics
Settings

RSI > KERNEL > Back Common Libs


Back Common Libs 
Project ID: 136

20 Commits 2 Branches 0 Tags 287 KB Files 1.4 MB

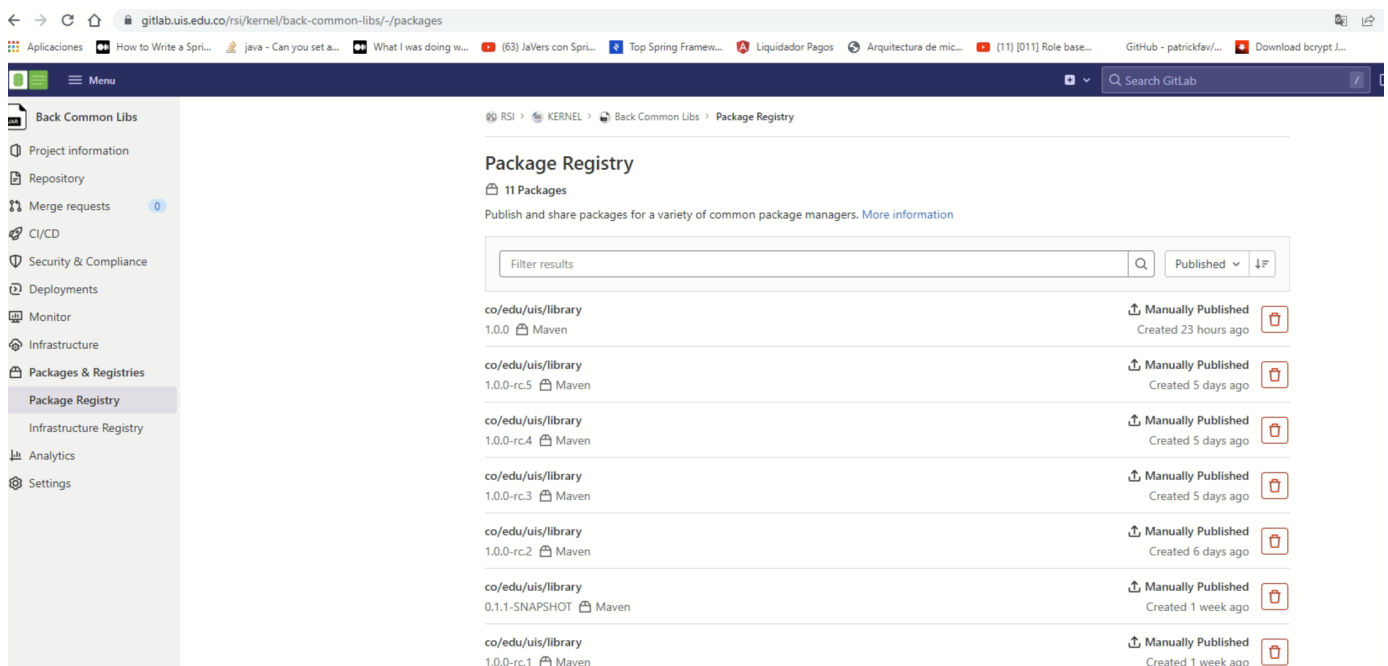
predev back-common-libs / +

Merge branch 'feature/first-version' into 'predev' 
Diego Vergel Paba authored 23 hours ago

Package Registry
Infrastructure Registry

No license. All rights reserved 



Name	Last commit
------	-------------
















gitlab.uis.edu.co/rsi/kernel/back-common-libs/-/packages

RSI > KERNEL > Back Common Libs > Package Registry

Package Registry
11 Packages
Publish and share packages for a variety of common package managers. [More information](#)

Filter results  Published 

co/edu/uis/library 1.0.0 	Manually Published Created 23 hours ago 
co/edu/uis/library 1.0.0-rc5 	Manually Published Created 5 days ago 
co/edu/uis/library 1.0.0-rc4 	Manually Published Created 5 days ago 
co/edu/uis/library 1.0.0-rc3 	Manually Published Created 5 days ago 
co/edu/uis/library 1.0.0-rc2 	Manually Published Created 6 days ago 
co/edu/uis/library 0.1.1-SNAPSHOT 	Manually Published Created 1 week ago 
co/edu/uis/library 1.0.0-rc1 	Manually Published Created 1 week ago 