2. Llevando IIC2233 a tu PC gracias a `git`

El objetivo de esta parte es tener un primer contacto con `git`, el sistema de control de versiones que se utiliza en este curso.

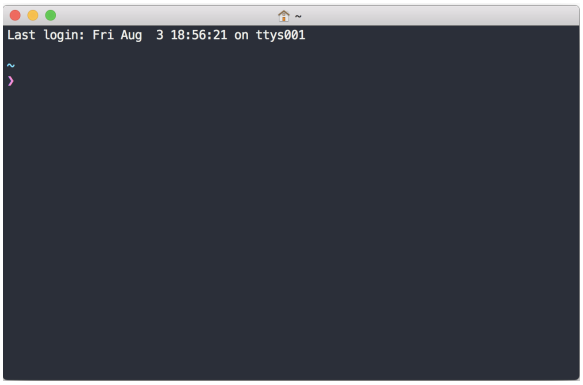
Abrir la terminal

Necesitamos aprender a navegar por las carpetas del computador usando la consola. En esta parte de la guía, sigue todos los pasos y verifica que te sale lo mismo que está aquí.

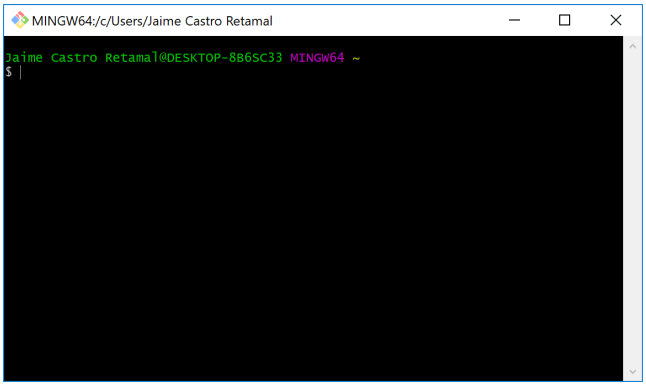
Para abrir la consola:

- **macOS o Linux:** busca el programa `Terminal` o similar.
- **Windows:** busca el programa `Git Bash`. Este programa es una consola que además implementa algunos de los comandos que podrías encontrar en Linux, por lo que es mucho más amigable que el "Símbolo del Sistema".

La consola en todo momento está ubicada en una carpeta (o directorio). Al abrir la consola, ésta se abre en la carpeta de tu computador que contiene tus datos. Esta carpeta suele llamarse `home` o "∼". Dentro de ella, suele encontrarse el escritorio o la carpeta de documentos.



(a) Consola abierta en macOS



(b) Consola abierta en Windows

Luego, puedes usar el comando `cd` para moverte por diferentes carpetas. Recomendamos revisar los contenidos de terminal de la semana 1 para aprovechar lo que más puedas la terminal. ¡Esta la usarás durante todo el semestre!

Clonar repositorio personal

En esta parte, clonaremos tu repositorio personal para que puedas entregar tus actividades y tareas. Para esto, primero necesitamos generar un *token* especial, que nos servirá como contraseña para poder acceder a cualquier repositorio que no sea público, por ejemplo, tu repositorio personal.

Para esto, primero debes acceder a este enlace https://github.com/settings/tokens/new y ingresar con tu cuenta de Github. Verás una vista como la Figura 4.

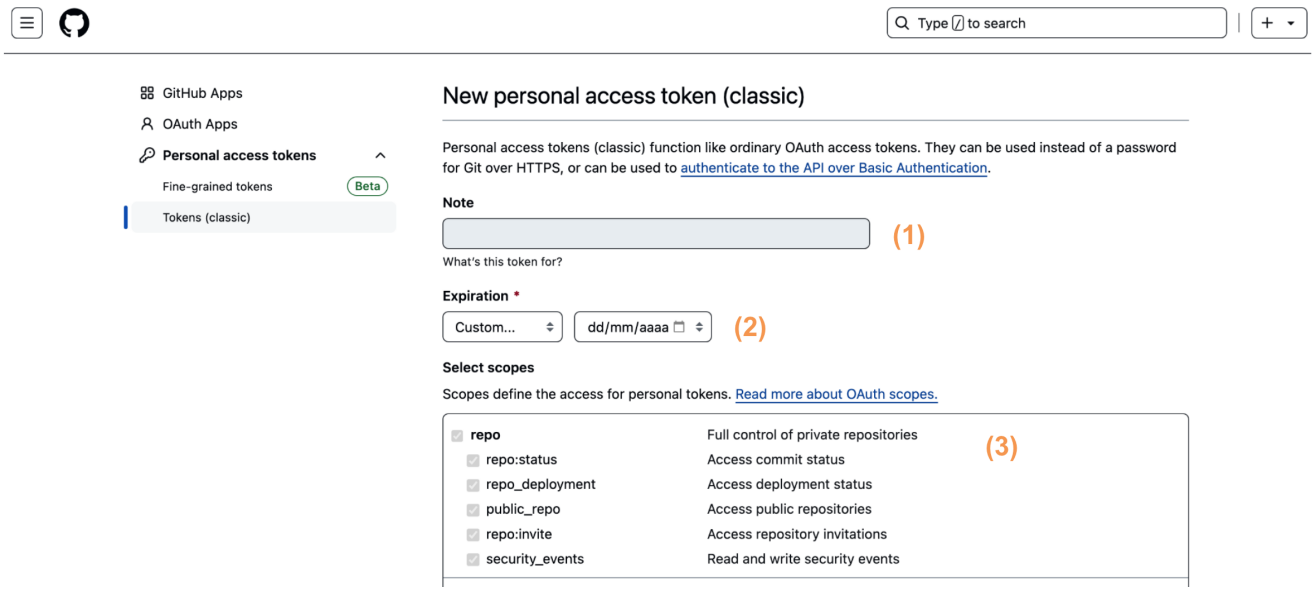


Figura 4: Interfaz para generar *token* personal

Luego, debes completar este formulario del siguiente modo:

1. Agregar una descripción en la sección de *Note*. Este será un texto para darle una descripción al *token* generado.
2. Definir una fecha de expiración. Se recomienda poner una posterior al cierre de semestre. Una vez pase la fecha definida, el *token* ya no servirá y deberás generar uno nuevo.
3. En la sección de *Select scopes*, haz *click* en cada cuadrado. Así haremos un *token* con todos los permisos posibles. Más adelante, en la carrera, aprenderás a generar un *token* con permisos más exclusivos.

Tras cumplir con los 3 pasos, debes hacer *click* en un botón que está al final del formulario que dice ***Generate token***. La vista de tu navegador se actualizará y te aparecerá un texto que comienza con `"ghp_"`. Ese será tu *token* que deberás guardar y usar siempre en tu terminal para interactuar con tu repositorio privado (*clone, pull, push*). No pierdas este *token* porque solo aparecerá una vez al momento de generarlo. En caso de olvidarlo, deberás crear uno nuevo siguiendo todos los pasos anteriores.

Ahora que ya tienes tu *token* personal generado. Podemos empezar con el proceso de clonar tu repositorio.

1. Asegúrate de que la terminal esté dentro de la carpeta donde quieras tener tu repo, como el escritorio o alguna carpeta propia. Si no, navega usando los comandos mencionados hasta encontrarlo.
2. Ve a la página de tu repositorio y copia el vínculo que permite clonar el repositorio (busca el botón verde que resalta y que muestra la figura 5).
3. En la consola, ejecuta el comando para clonarlo: `git clone url_copiada`.
4. Se te pedirá ingresar tu usuario y luego el *token* que acabamos de generar. Este **no se verá en tu terminal al momento de copiar**, pero es normal porque es por temas de seguridad. Luego presionas *enter* y se comenzará a clonar tu repositorio.
5. Deberías ver que se creó la carpeta con el contenido de tu repositorio personal[1].

[1]Si no sabes dónde se creó, recuerda revisar los contenidos de terminal, hay el comando para indicar tu posición

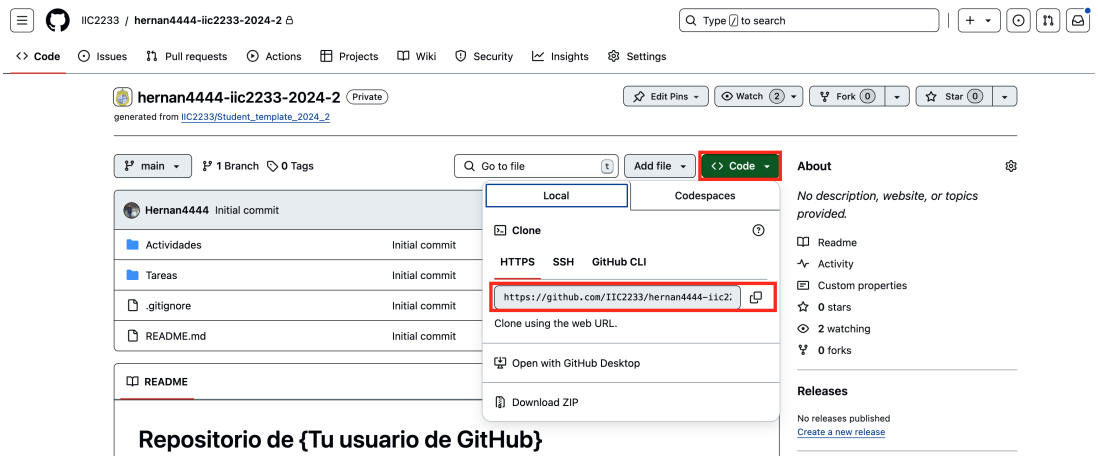


Figura 5: Donde encontrar enlace de repositorio.

En caso de que ya hayas copiado tu repositorio personal en un lugar que no te guste, simplemente puedes mover la carpeta completa a un directorio que prefieras, los archivos que permiten que `git` funcione se encuentran dentro de tu carpeta y seguirán funcionando normalmente.

Clonar otros repositorio importantes del curso

Aprovechando el vuelo, repite los mismos pasos del punto anterior pero para `Syllabus` y para `contenidos`, ya que tendrás que estar pendiente de estos repositorios durante el semestre.

El primer *add*, *commit* y *push*

Haremos nuestro primer *commit* dentro del repositorio personal. Para trabajar en él, tu consola deberá estar **dentro** de la carpeta del repositorio (podrías usar el comando `cd`).

1. Dentro del repositorio, hay un archivo llamado `README.md` creado. En este, hay información sobre el uso de tu repositorio, y dejamos espacios para que rellenes con tus datos. Abre el archivo en tu computador con algún editor de texto y rellena los espacios con tus datos.
2. Ahora vuelve a la consola. Revisa el estado del repositorio utilizando `git status`. Observa que tu archivo recién editado aparece listado como *"modified"* en *"Changes not staged for commit"*.
3. Agrega el archivo `README.md` al *stagging area* mediante el comando: `git add README.md`.
4. Revisa el estado del repositorio (`git status`) y verifica que tu archivo recién editado aparece listado como *"modified"* en *"Changes to be committed"*.
5. Utiliza: `git commit -m "README actualizado"`. El texto entre comillas se conoce como mensaje del *commit* y debe describir lo que fue agregado mediante `git add`.
6. Vuelve a revisar el estado (`git status`) del repositorio nuevamente. No deberías ver listado a `README.md`, pero si un mensaje que dice *"Your branch is ahead of 'origin/main' by 1 commit."*, lo cual significa que el *commit* fue creado correctamente,
7. Ahora, para escribir tus cambios en GitHub utiliza: `git push`.
8. Vuelve a revisar el estado del repositorio (`git status`), debe decir: *Your branch is up to date with 'origin/main'. nothing to commit, working tree clean.*