**¿Qué es GitHub?**

Se puede entender como una palabra compuesta: **Git** y **Hub**.

**¿Qué es Git?**

Una herramienta que facilita la colaboración:

Imagínate que tú y Beto deciden dibujar juntos en la escuela, pero quieren terminar el dibujo cada quien en su casa. La maestra saca 2 fotocopias de lo que hicieron en clase y se la llevan a casa (**clone | pull**).

Tanto como tú y Beto le añaden cosas al dibujo (**add**), y a cada cosa que añaden le ponen una nota explicando lo que agregaron (**commit**).

Cuando regresan a la escuela, la maestra corta y pega los dibujos en uno solo (**merge**), si ambos colorearon la misma área (**merge conflict**) alguien decide qué versión se usa, si la de Beto o la tuya.

Y como son fotocopias, pueden hacer los cambios que quieran sin dañar el original.

Si deciden ver *quién hizo qué cambio y por qué*, sólo basta con ver las notas que anexaron tú y Beto (**log**). Y hasta pueden comparar las 2 versiones (**diff**).

Si alguien más quiere dibujar con ustedes, sólo necesita una copia del original (**clone**).

Ese sería un flujo de trabajo con **Git**. Las palabras que están en **negritas** es el nombre en inglés que se le da a esa acción.

*Si son de los que olvidan los comandos de Git, aquí los pueden checar:* [Hoja de comandos de Git en español](https://github.com/ArslanBilal/Git-Cheat-Sheet/blob/master/other-sheets/git-cheat-sheet-es.md)

*Y si quieres meterle esteroides a tu Git, checa este enlace:* [Git Extras](https://github.com/tj/git-extras)

*Por favor, traten de usar la guía de estilo:* [Guía de estilo de Git](https://github.com/agis-/git-style-guide)

**¿Y el *Hub*?**

Un espacio *en línea* para tener un respaldo de tus proyectos.

**¿Entonces...?**

**GitHub** es *un espacio en línea para tener un respaldo de tus proyectos* que usen *Git*, enfocado a **proyectos de programación con código abierto**.

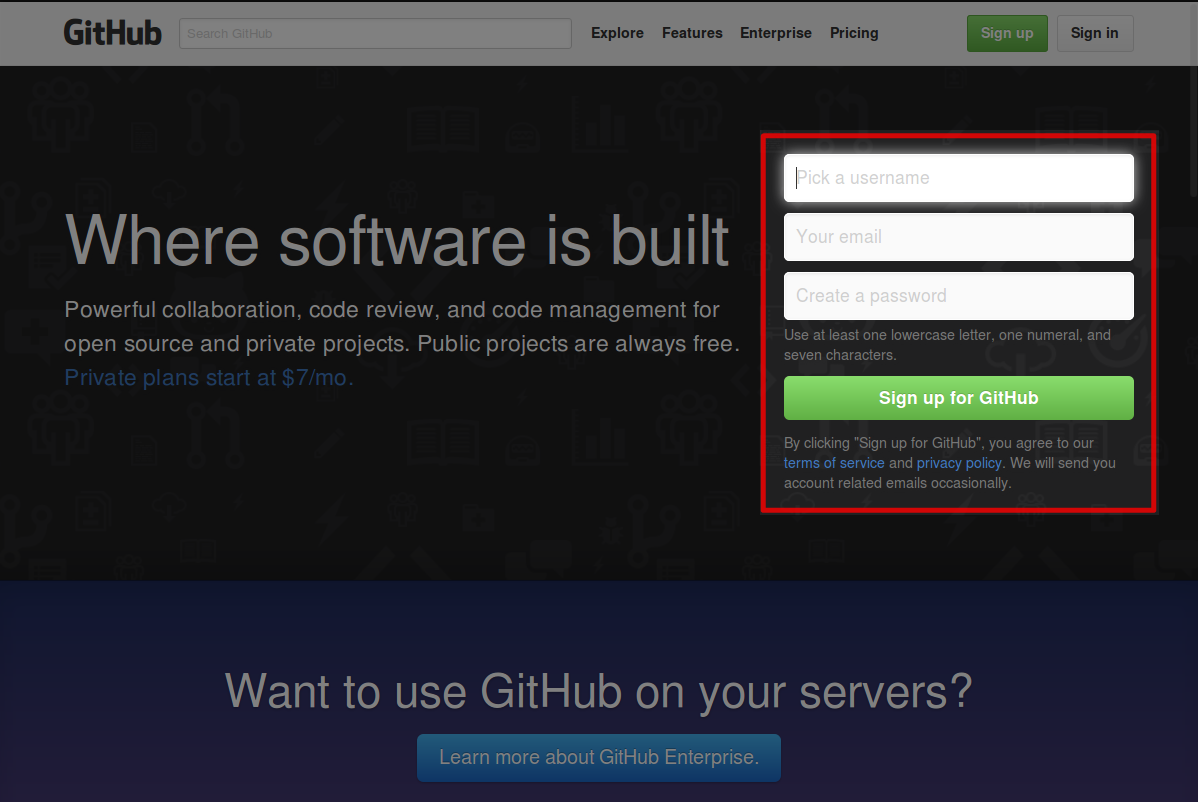
Cuando digo *código abierto* me refiero a que la gente pueda ver cómo está construido el proyecto (ver el código fuente).

Los proyectos están organizados en **repositorios**. Es un lugar donde se almacenan los archivos que conforman el proyecto.

Crear una cuenta en **GitHub** es gratuito, siempre y cuando tus proyectos sean abiertos al público. Si quieres restringir el acceso a tus proyectos puedes pagar por ello una subscripción mensual.

**¿Cómo hago una cuenta?**

Ingresa a la página principal de [Github](https://github.com/) y en los campos de texto llena tu nombre de usuario, correo electrónico, y contraseña.

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-home.png)

**¿Y ahora qué hago?**

Si ya tienes un proyecto en el que estés trabajando, ve al repositorio. El enlace se debe de ver algo así: http://github.com/usuario/nombre-del-repositorio O puedes [crear un repositorio](https://github.com/MrOutis/GitHub-Simple#crea-un-repositorio)

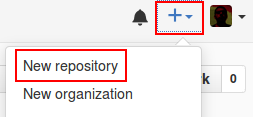
**Repositorios**

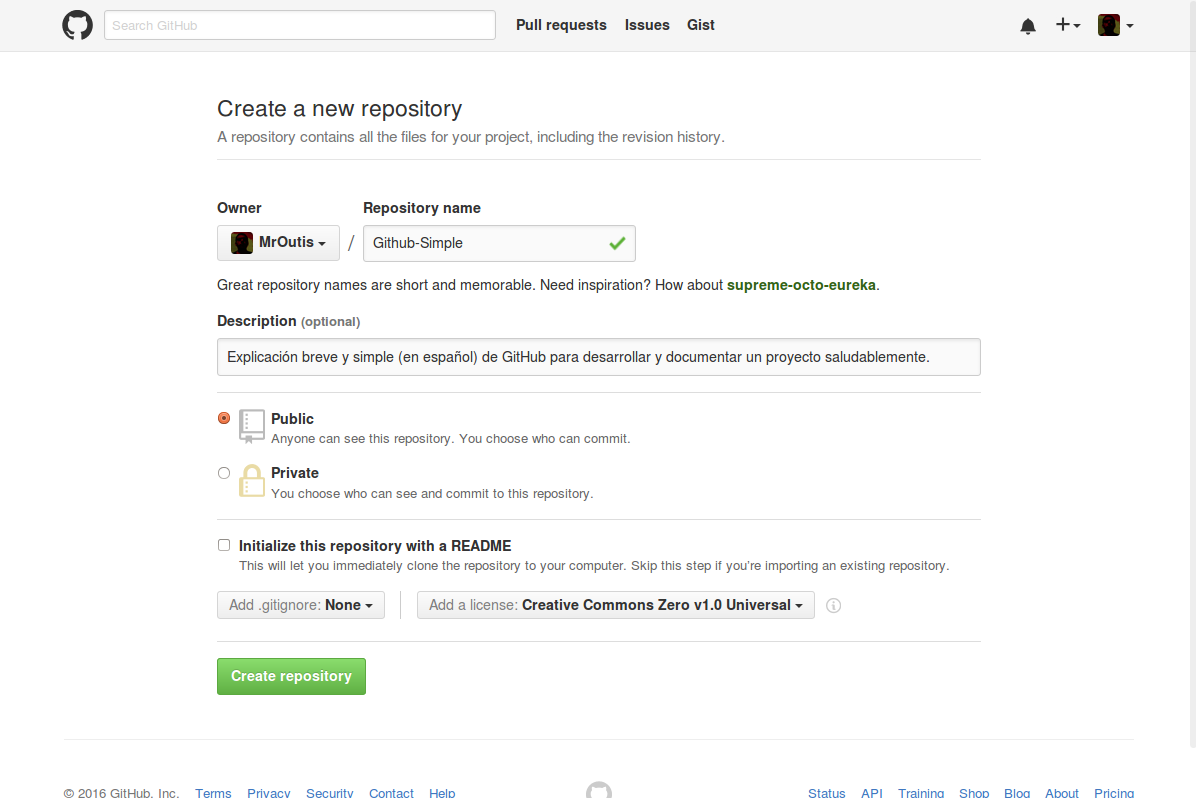
Es el lugar donde están todos los archivos del proyecto, incluyendo la documentación. Los repositorios pueden tener múltiples colaboradores y pueden ser **públicos** (que el acceso esté **abierto** a cualquiera) o **privados** (**restringe** el acceso a las personas que autorices).

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-repository.png)Un repositorio se ve así:

Repasaremos la interfaz del repositorio, empezando de arriba hacia abajo. Pero primero, crearemos uno.

**Crea un repositorio**

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-new-repository.png)En la esquina superior derecha, dale click en el ícono de +, seguido de New repository:

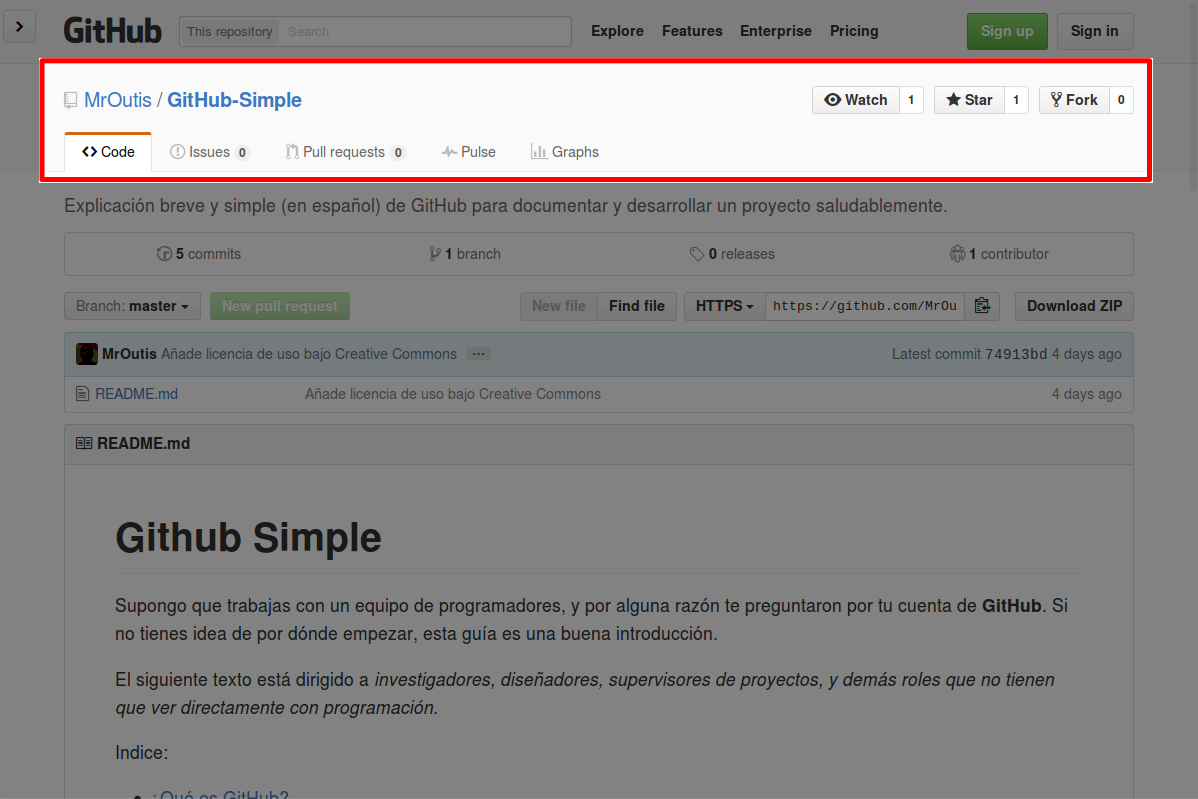
Y te va a llevar a una página similar a esta: [](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-create-repository.png)

Vienen varios campos que tienes que llenar:

* **Owner**: El dueño del repositorio (puedes ser tú o alguna organización a la que pertenezcas).
* **Repository name**: Un nombre corto y memorable para identificarlo.
* **Description**: Descripción breve del contenido.
* **Public / Private**: Si el acceso eso **abierto** o **restringido**.
* **Initialize with README**: Añade un [README](https://github.com/MrOutis/GitHub-Simple#readme) al repositorio.
* **Add .gitignore**: Es un archivo de texto que le dice a **Git** qué archivos ignorar. Por ejemplo, si no quieres tener archivos *PDF*, en el .gitignore escribes \*.pdf, o si tienes una carpeta que se llama Pruebas y no la quieres incluir, escribes Pruebas/.
* **Add a license**: Sólo por que tus archivos estén en internet, no significa que sean libres de usar o de dominio público. Esta opción genera un archivo que se llama **LICENSE**, y deja en claro qué se puede hacer con los archivos y qué no.

*Si tienes duda respecto a las licencias de uso, visita:* <http://choosealicense.com/>

**Encabezado**

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-repository-header.png)

Contiene el **propietario** y el **nombre** del repositorio.

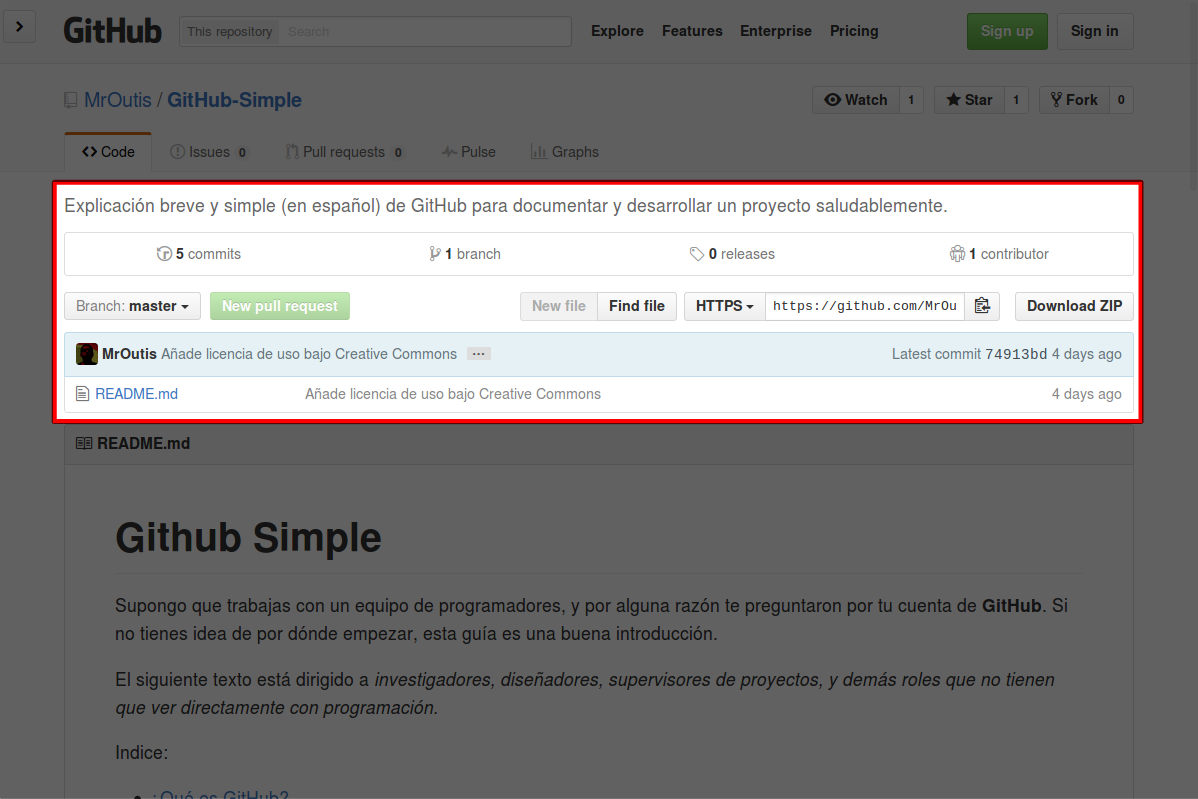
También hay tres botones que representan acciones:

* [Watch](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-watch.png): Recibes notificaciones cuando hay algún cambio en el proyecto. Sirve para estar al día.
* [Star](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-star.png): Se utilizan para tener **fácil acceso** a un proyecto, y para mostrarle tu amor al repositorio, similar a los **likes**en Facebook 👍. *Si quieres ordenar tus Stars puedes utilizar [AstralApp](http://astralapp.com/)*
* [Fork](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-fork.png): Haces una **copia** del repositorio en tu cuenta de GitHub *(recuerda las fotocopias que se llevaron Beto y tú)*, así puedes hacer cambios sin afectar el original. También te permite **sugerir cambios** y **traer actualizaciones** del original a tu *fork*. Se utilizan para **iterar** sobre **ideas** e **implementaciones** manteniendo el original intacto.

[Barra de navegación en GitHub](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-header-navbar.png)Y una barra de navegación:

Vamos a explorar cada una de las pestañas, empezando por **Code**.

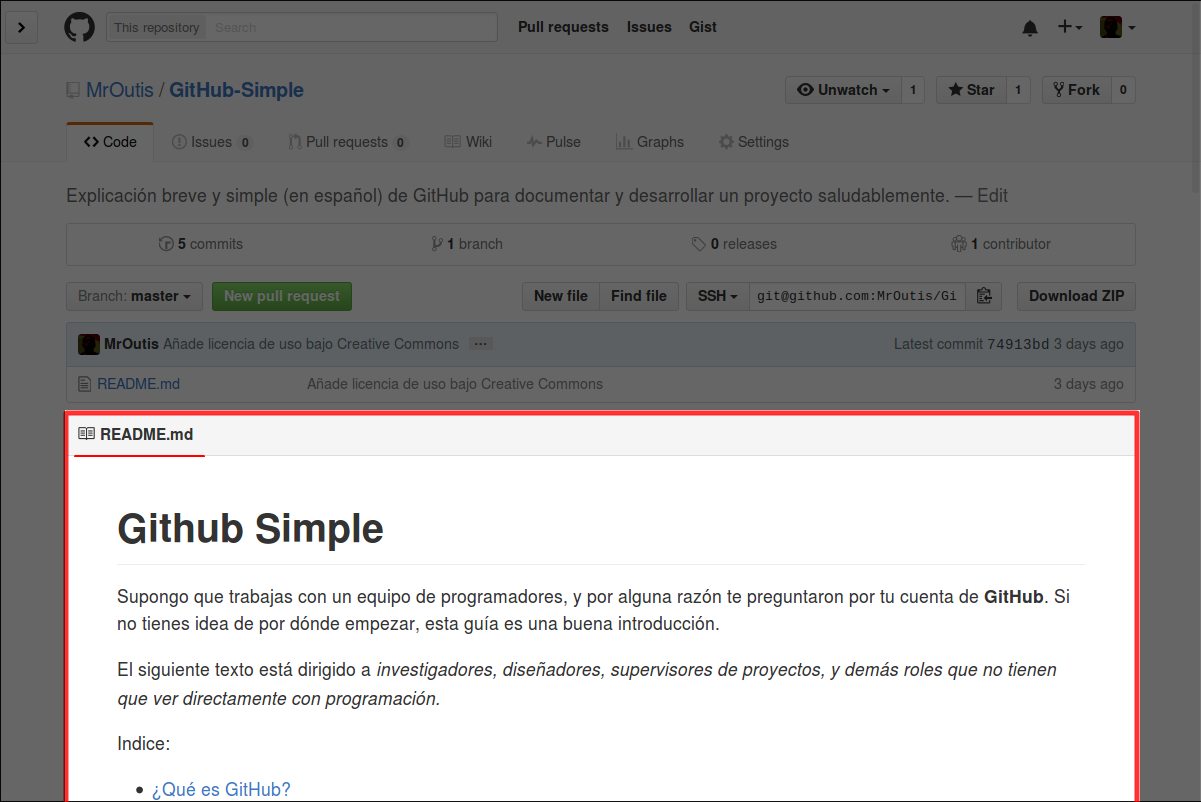
**Panel de Código**

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-code-panel.png)

Contiene una descripción corta del proyecto. Algunas métricas y acciones de **Git**. Y un listado de archivos que contiene el repositorio.

**README**

Es el documento que da la **bienvenida** a tu proyecto.

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-readme.png)

Se traduce del inglés como **LEEME**.

Sirve como **introducción**, y es un buen lugar para **explicar** o **vender** tu idea. Si quieres que la gente **utilice** y/o **contribuya** a tu proyecto, aquí debes de convencerlos.

Normalmente las personas ven el README en un navegador, y para que no se vea como un simple texto, se le agrega formato utilizando [Markdown](https://es.wikipedia.org/wiki/Markdown) (de ahí viene la extensión de archivo .md).

**Desarrollo guiado por README**

El README es la [piedra angular](https://es.wikipedia.org/wiki/Piedra_angular) de un proyecto.

Si empiezas por escribir un documento sencillo que sirva como introducción, quien colabore podrá tener siempre presente el **¿Por qué?** y **¿Para qué?** del proyecto, mejorando la **comunicación** y respaldando que todos estén apuntando a la misma dirección.

Te ayudará a conocer **qué implementar** para cumplir los ideales del proyecto.

Además, es un esquema bastante sencillo para **aterrizar las ideas**. Facilita que se pueda **discutir** e **iterar** sin perder el rumbo.

**Contenido de un README**

Un buen README debe tener al menos estos elementos:

* **Descripción**: ¿De qué trata el repositorio? ¿Por qué me debería importar?
* *Sociales*: (Opcional) Si tu proyecto tiene algún Blog, Twitter, etc. inclúyelos
* **Instrucciones de uso**: Proceso de instalación, comandos, interfaz gráfica, etc.
* **Dónde obtener ayuda**: Enlaces a documentación, Wiki, Foros, etc.
* **Guía de contribución**: Especifica formato del código, convenciones, cómo abrir un ticket para seguimiento de problemas, cómo presentar los cambios, etc. Normalmente se encuentra en un archivo aparte, nombrado **CONTRIBUTING**. La función de esta sección es crear orden y facilitarle el trabajo a quien mantiene el repositorio.
* **Lista de contribuidores**: Las personas que están atrás del proyecto. Si son demasiadas, inclúyelas en un archivo aparte **CONTRIBUIDORES**. *Puedes utilizar esta herramienta para darles un buen formato:* [Generador de tabla de contribuidores](https://github.com/stoeffel/gh-contributors-table)
* **Fuentes**, **inspiración**: Da **reconocimiento** si utilizaste ideas o material de otros proyectos. También les ayuda a los contribuyentes a tener una perspectiva más amplia.
* **Licencia de uso**: En la [creación de un repositorio](https://github.com/MrOutis/GitHub-Simple#crea-un-repositorio) hablamos al respecto. Esta licencia puede ir en un archivo llamado **LICENSE** o incluido en el README. *Si está en LICENSE no olvides mencionarlo en el README*.

**Ejemplos de READMEs**

Una lista de READMEs para ejemplificar casos de buenas prácticas.

* [Deep Belief SDK](https://github.com/jetpacapp/DeepBeliefSDK): Descripción, tabla de contenido, instrucciones de uso muy claras, instalación, ejemplos de uso, preguntas frecuentes, licencia, créditos.
* [twtxt](https://github.com/buckket/twtxt): Descripción, demostración en video, tabla de contenido, instrucciones de uso muy claras, contribuciones, licencia.
* [GrapeJS](https://github.com/artf/grapesjs): Descripción, características con imágenes, instrucciones de uso muy claras, instalación, contribución, licencia.
* [PHP brew](https://github.com/phpbrew/phpbrew): Descrpición, traducciones (chino), instalación, instrucciones de uso muy claras, guía de contribución, documentación, licencia.
* [spf13-Vim](https://github.com/spf13/spf13-vim): Descripción, instalación, imágenes, instrucciones de uso muy claras, licencia.
* [Underscore CLI](https://github.com/ddopson/underscore-cli): Descripción, instalación, ejemplos con imágenes, instrucciones de uso muy claras, cómo reportar erroes, inspiración, licencia.

**Gestionar un repositorio**

La mayoría de los problemas de nuestro trabajo no son tanto técnicos como sociológicos en su naturaleza.

Uno de los retos más grandes de un proyecto *open source*, es lidiar con las personas. Cuando facilitas la comunicación, haciendo **explícitas** las **tareas** y **metas** que se quieren cumplir, no suele ser tan complicado.

Para esto, hay metodologías que se pueden seguir, la más utilizada en el desarrollo de software es la ágil. **Te recomiendo leer más sobre desarrollo ágil, ya que te ayudará a organizarte mejor con tu equipo**

Una vez que escribes una definición del proyecto ([README](https://github.com/MrOutis/GitHub-Simple#readme)), puedes empezar a definir **tareas** y **metas** explícitas.

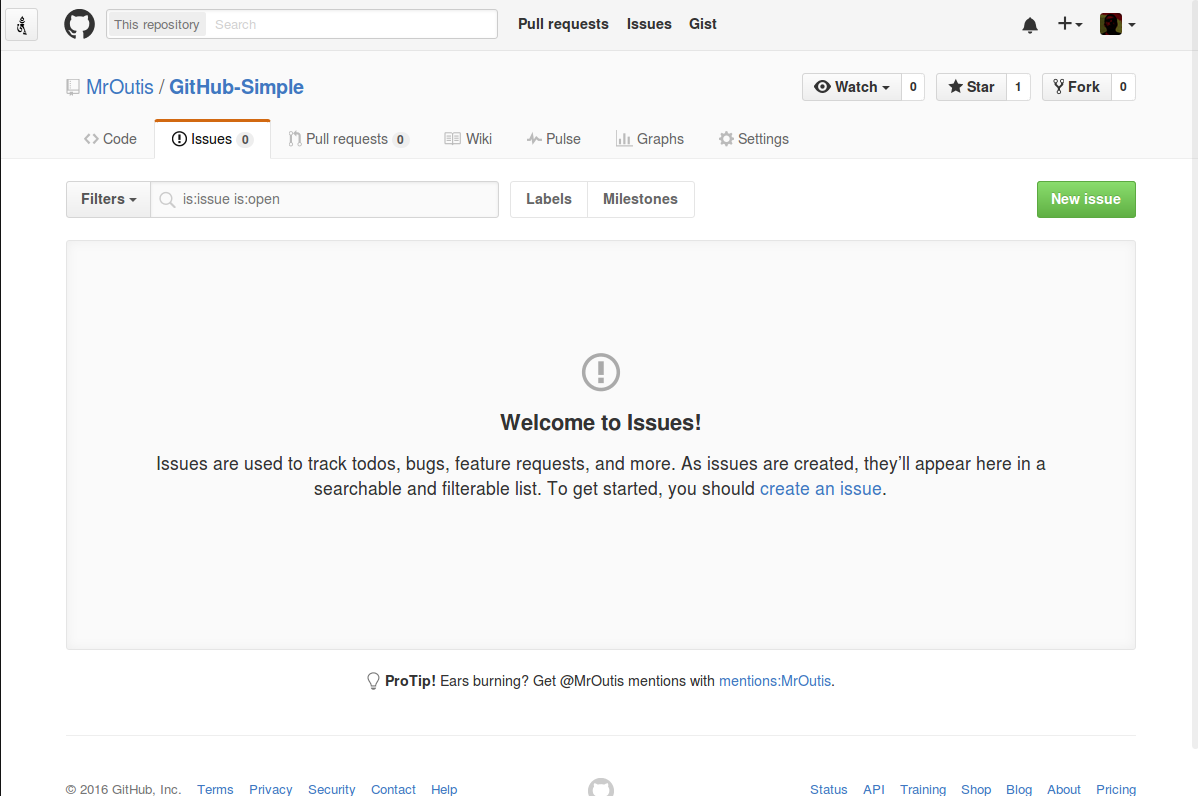
Algunas herramientas de **GitHub** te permiten tener todo centralizado:

* **Issues**: Tareas que se necesiten cumplir, ya sea arreglar un problema o agregar una mejora al proyecto.
* **Labels**: Etiquetas para categorizar las tareas
* **Milestones**: Metas u objetivos, están compuestos de muchas tareas.
* **Pull requests**: **Petición de algún cambio**, ya sea agregar o quitar información del repositorio (archivos, líneas de código, arreglar errores, implementar mejoras, etc.)

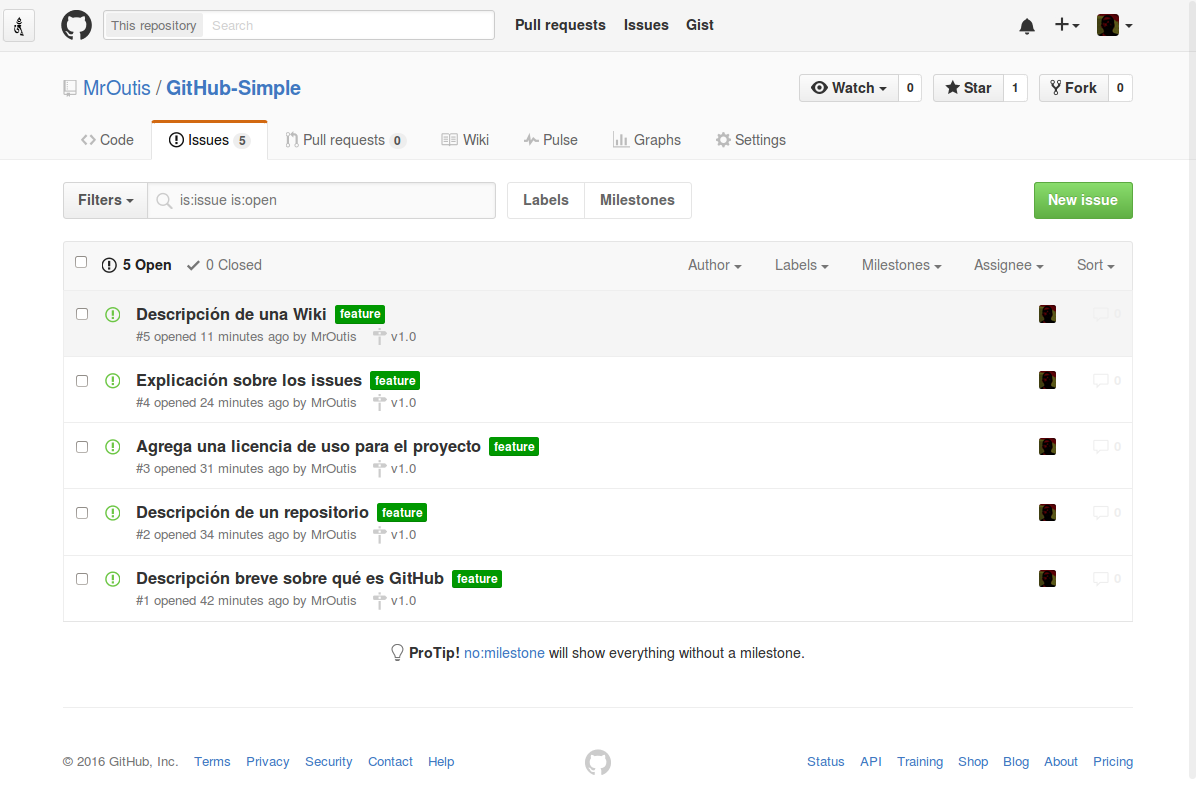
**Issues**

[Github issues tab](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-header-navbar-issues.png)Si le damos clic en la pestaña de *Issues*:

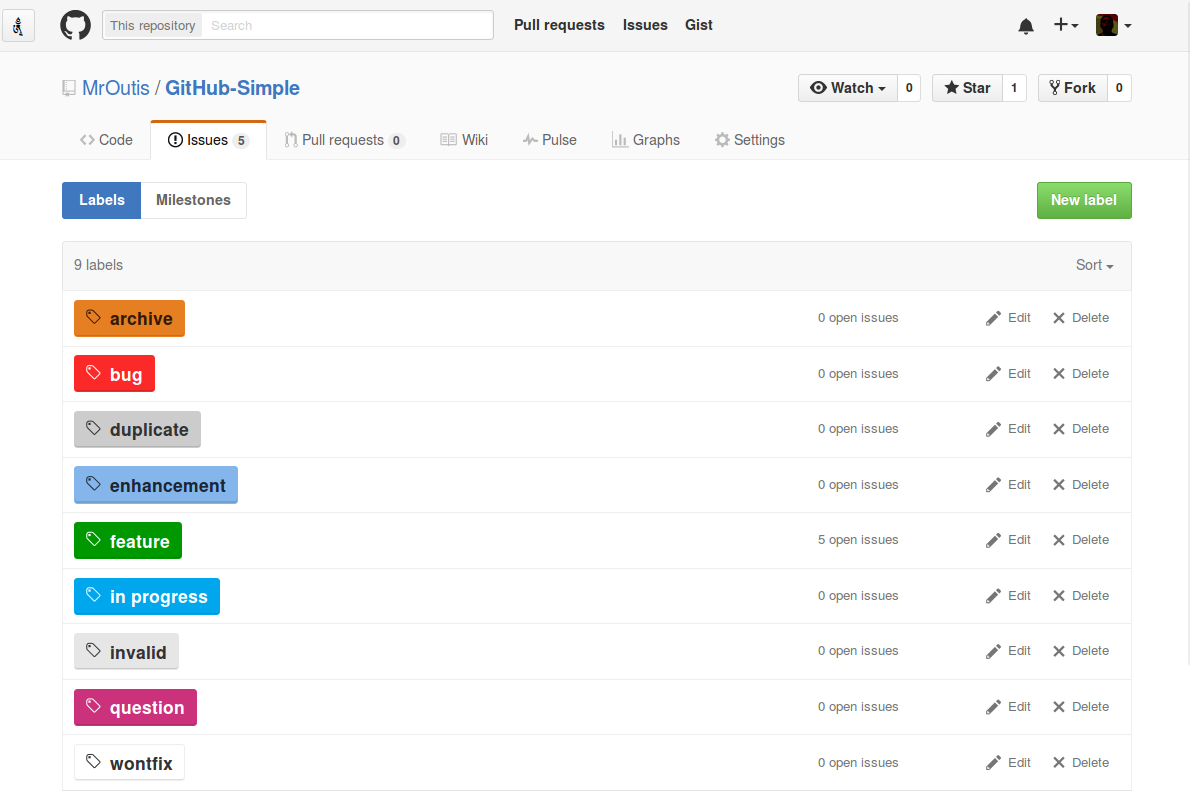
Nos aparecerá una página similar a esta:

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-issues.png)

Ahí pueden empezar a crear tareas. La interfaz es muy simple, sólo recuerden ser **explícitos** a la hora de nombrar y describir las tareas. Estas pueden ser [historias de usuario](https://es.wikipedia.org/wiki/Historias_de_usuario), necesidades del proyecto, etc. Para seguir un estandar, se utiliza la **Guía de contribución**. Tengan en cuenta a quién va dirigida la tarea, incluye referencias explicando **qué quieres** y **cómo lo quieres**.

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-open-issues.png)Después de que hayan escrito algunas, se verán algo así:

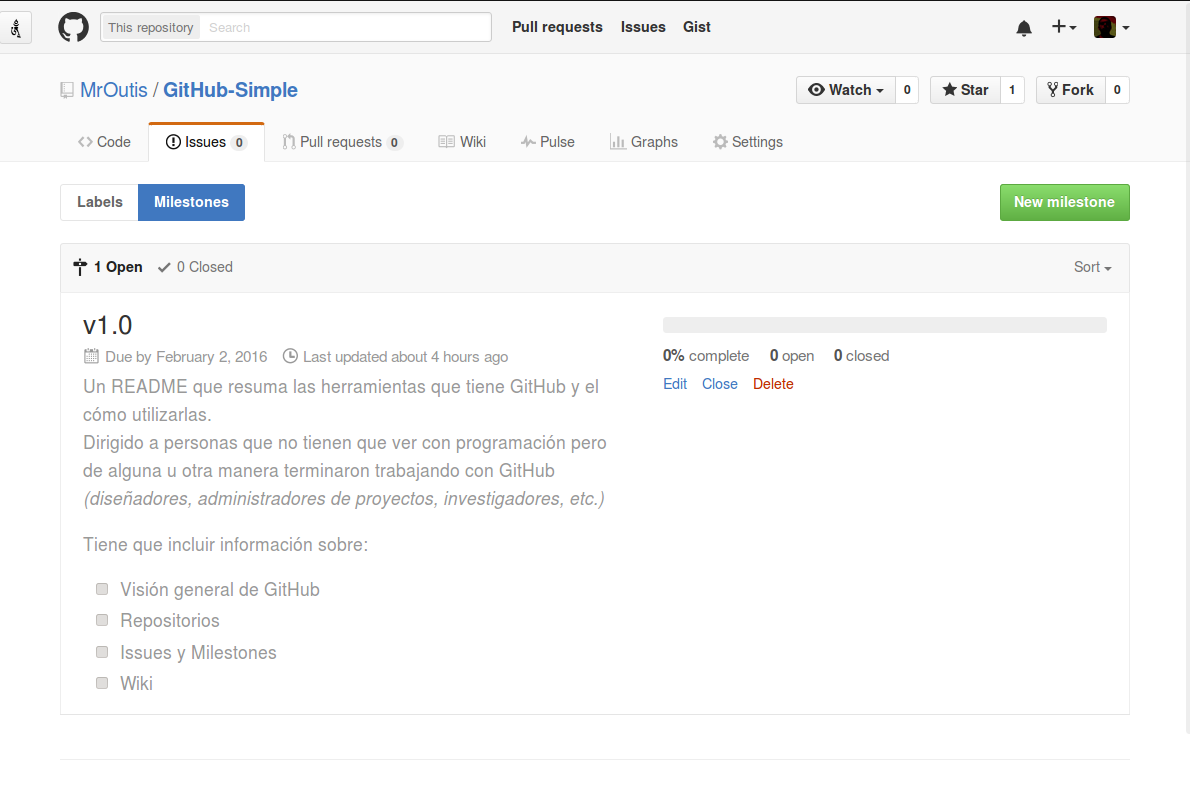
**Labels**

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-labels.png)Se traduce al español como **etiquetas**, sirve para categorizar tareas. La pantalla se ve algo así, ahí pueden agregar, editar, y eliminar etiquetas:

**Milestones**

En sí, es un conjunto de tareas. Puede ser un lanzamiento, un módulo del proyecto, una revisión, etc. Incluyen una **descripción**, **fecha**, y las **tareas** que se necesitan cumplir para lograr ese objetivo.

Se puede decir que los **issues** son metas a corto plazo y los **milestones** son a largo plazo.

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-milestones.png)La pantalla de los milestones se ve algo similar:

**Herramientas alternativas**

Existen herramientas que te pueden ayudar a administrar mejor tu repositorio. Si estás acostumbrado a utilizar un [Kanban](https://es.wikipedia.org/wiki/Kanban) para tus tareas, puedes utilizar [Waffle.io](https://waffle.io/).

Puedes encontrar más integraciones de GitHub en: <https://github.com/integrations>

*El consumo excesivo de integraciones puede ser dañino para la salud de un proyecto. Integra con medida.*

**Documentación**

Estarás usando tu c­­ódigo dentro de 6 meses.

La mejor razón para escribir documentación, es porque **tenemos mala memoria**. Cuando queramos retomar el proyecto, se nos habrá olvidado la mayoría de las cosas. *Sí tan solo hubiera escrito el porqué de esto, la vida sería más simple*.

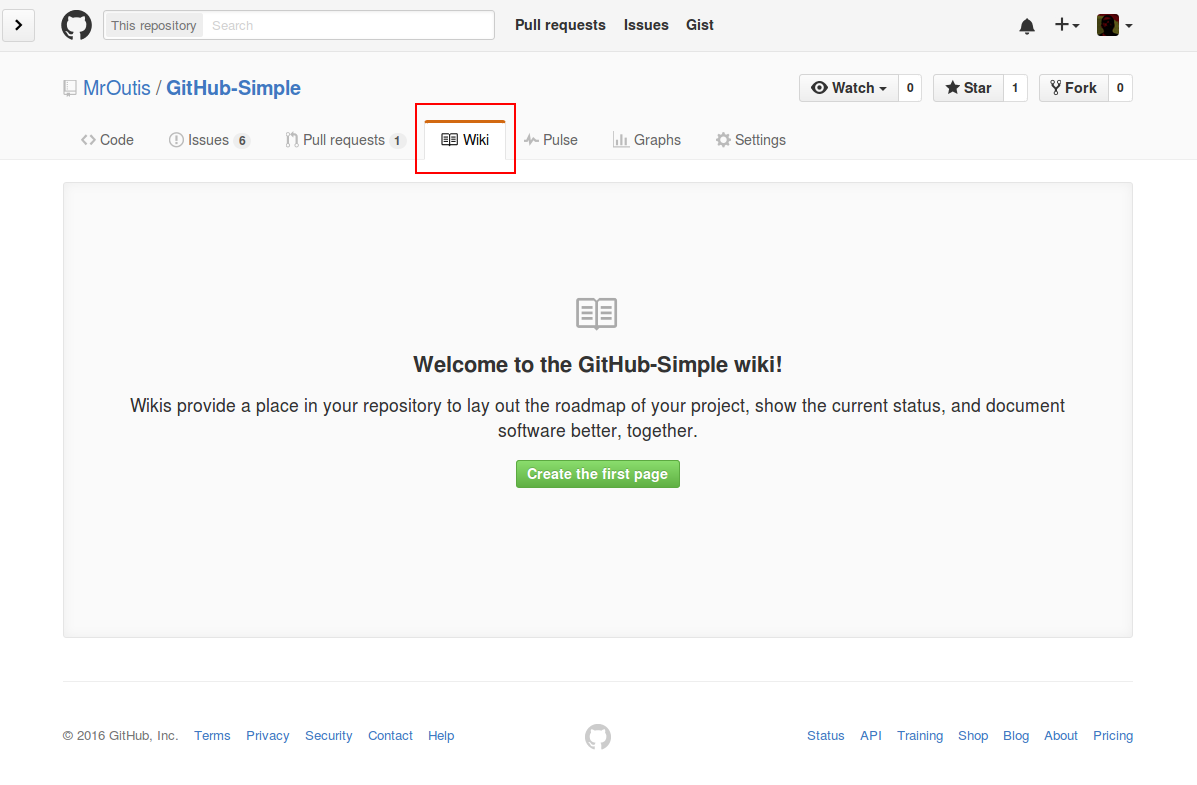
**La documentación expresa lo que hay detrás del código**. Se enfoca en el **por qué** en lugar del cómo. Sí las personas no saben el **propósito** de tu proyecto, no lo usarán ni serán contribuyentes.

La primera documentación que se debe escribir es el [README](https://github.com/MrOutis/GitHub-Simple#readme).

Una vez que esté más estructurado el proyecto, puedes expandir la documentación a una [Wiki](https://github.com/MrOutis/GitHub-Simple#wiki).

**Wiki**

Es una de las herramientas que provee GitHub. Un lugar centralizado para documentación.

[](https://github.com/mroutis/GitHub-Simple/blob/master/images/github-wiki.png)Está compuesta de páginas, como un libro.