

[Inicio](#) | [Quiénes somos](#) | [Formación](#) | [Publicaciones](#)

Tutoriales.

[adictosaltrabajo](#) / [Tutoriales](#) / [Primeros pasos con Source Tree](#)**Juan Antonio Ortiz Carvajal**

Consultor tecnológico de desarrollo de proyectos informáticos.

Puedes encontrarme en [Autentia](#): Ofrecemos servicios de soporte a desarrollo, factoría y formación.

Somos expertos en Java/Java EE



Primeros pasos con Source Tree

junio 8, 2015

Juan Antonio Ortiz Carvajal

5 comentarios

Tutoriales

9537 visitas

En esta ocasión hablaremos de Source Tree, una herramienta tanto para Git como para Mercurial (aunque esta prueba es para Git) para trabajar con estos sistemas de versionado de forma gráfica.



0. Índice de contenidos.

- [1. Introducción](#)
- [2. Entorno](#)
- [3. Instalación](#)
- [4. Clonación de repositorio a local](#)
- [5. Subida de proyecto desde local al repositorio remoto](#)
- [6. Operaciones básicas de versionado con Source Tree](#)
- [7. Conclusiones](#)
- [8. Referencias](#)

1. Introducción

Una de las partes más importantes a la hora de realizar un proyecto consiste en tener un lugar centralizado en el que todo el equipo pueda acceder a la información del mismo. Para esto, el uso de un sistema de control de versiones (como GIT) es imprescindible.

2. Entorno

El tutorial está escrito usando el siguiente entorno:

Los más visitados de la semana

[Introducción a StencilJS y demo de integración con Angular](#)
[Javassist: manipulando el bytecode de una aplicación Java.](#)
[GIF animado con Photoshop](#)
[Page Analytics](#)
[Monitorizando equipos y servicios con Nagios + NagiosQL + PN...](#)


- Hardware: Portátil MacBook Pro 17' (2.8 GHz Intel Core 2 Duo, 8GB 1067 MHz DDR3, 256GB Solid State Drive).
- NVIDIA GeForce 9400M 256 MB
- Sistema Operativo: Mac OS X Yosemite Version 10.10.2
- GIT version 2.2.2 (Apple Git 55)

Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información


ACEPTAR

Tweets por @adictosaltrabaj

**adictosaltrabajo**
@adictosaltrabaj

TUTORIAL | Securizar un API

23min

**adictosaltrabajo**
@adictosaltrabaj

TUTORIAL | Introducción a #StencilJS y demo de integración con #Angular por @esquilera82

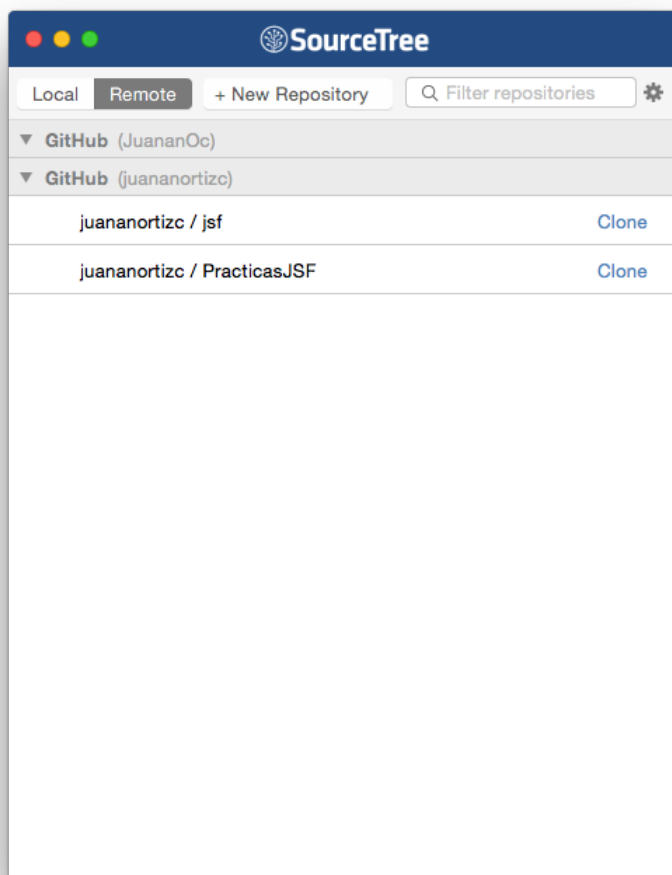
Insertar Ver en Twitter

3. Instalación

La instalación está cubierta en un tutorial previo realizado por Roberto llamado: [Integrando Xcode4 con GitHub](#)

4. Clonación de repositorio a local

Una vez instalado nos aparecerá una ventanita con las opciones básicas:



Donde podemos ver de izquierda a derecha:

- **Local:** Todos los proyectos que tenemos actualmente en local.
- **Remote:** Los proyectos que están actualmente en remoto, ordenados por repositorios (En la imagen se muestra esta pestaña, donde se ve como el primer repositorio no tiene ningún proyecto, mientras que el segundo tiene dos proyectos asociados).
- **+ New Repository:** Nos da la opción de crear un repositorio local o remoto, así como añadir un repositorio local existente, o clonar un repositorio remoto a local.
- Un **buscador**.
- **Boton de opciones**, donde podemos añadir nuevas cuentas y nuevos repositorios, así como actualizar la ventana.

Clonando el repositorio obtenemos una copia del mismo en local. Para realizarlo supondremos que tenemos un proyecto creado dentro de un repositorio. En nuestro caso se llama adictosTree.

El comando de git asociado es:

```
1 git clone [url]
```



Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR



JuananOc

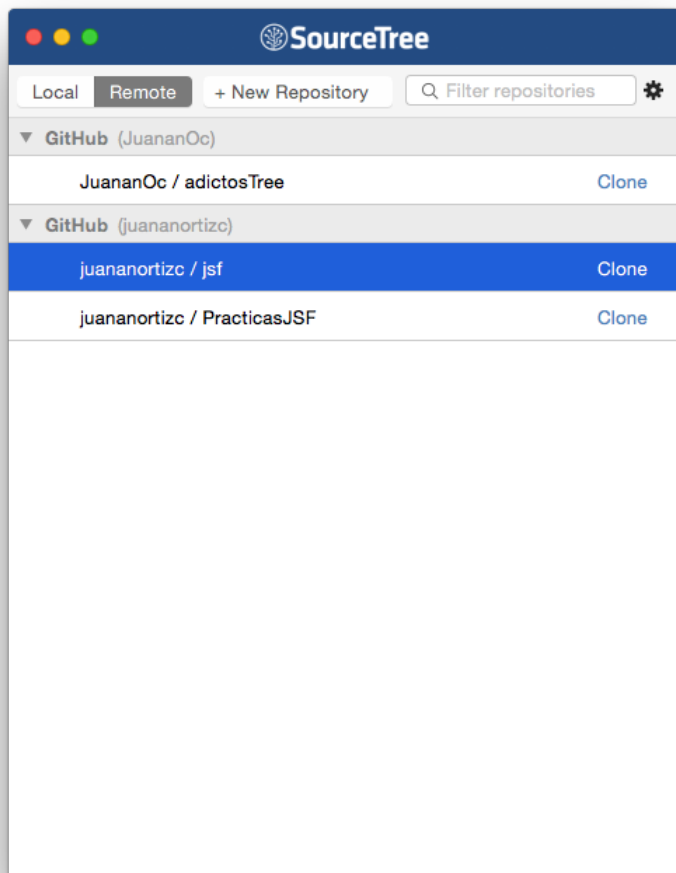
Popular repositories

adictosTree

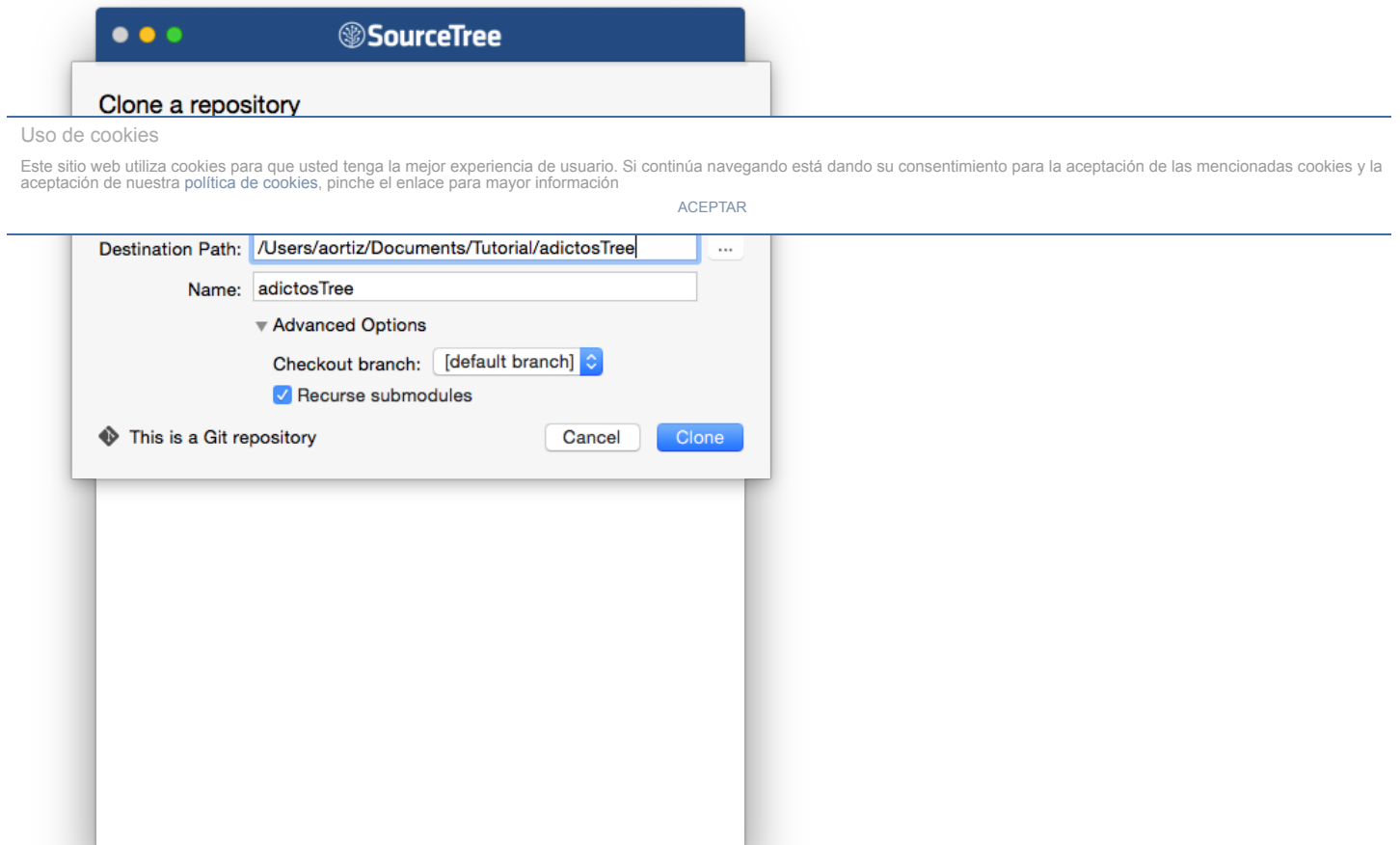
Repositorio para el tutorial de adictos al trabajo

0 ★

Actualizamos la ventana de Source Tree para que aparezca, en el botón de opciones -> Refresh



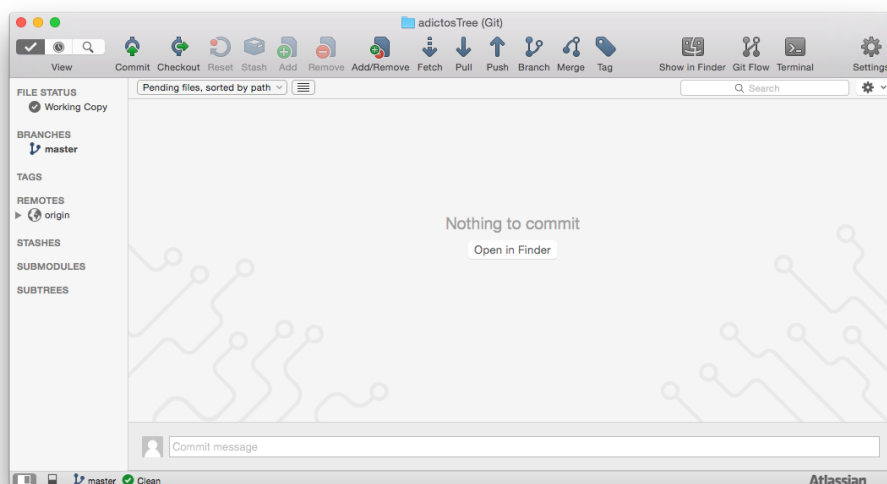
Como podemos observar, ahora aparece el proyecto que tenemos en remoto, pulsamos en clone y configuramos la ventana emergente:



Donde:

- **Source URL:** Es la ruta a tu repositorio(rellenada automáticamente)
- **Destination Path:** Ruta local en la que se clonará tu repositorio
- **Name:** Nombre del repositorio

En las opciones avanzadas puedes elegir la rama (branch) que quieres clonar a local. Al pulsar el botón clone, llegamos a la ventana principal de source tree. ¡Ya tenemos nuestro repositorio remoto clonado a local!



5. Subida de proyecto desde local al repositorio remoto

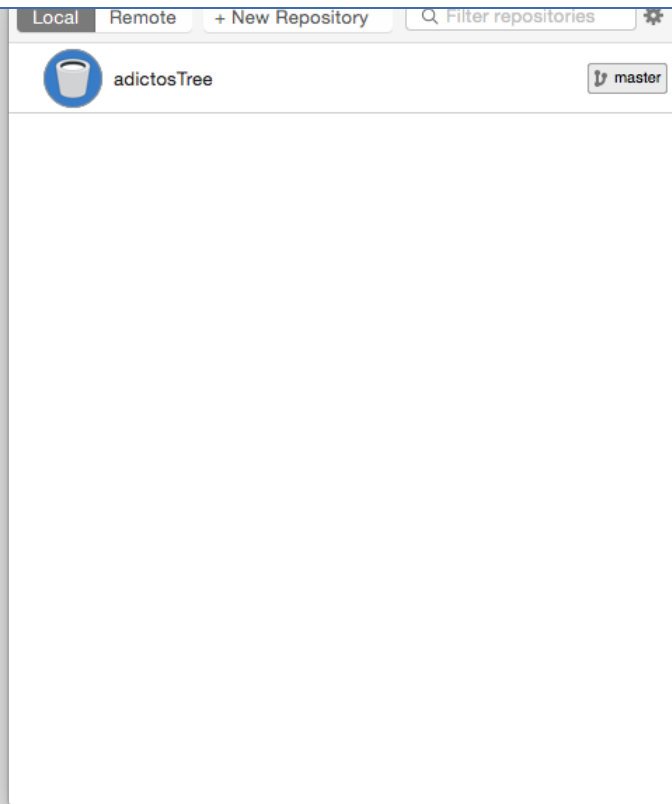
Pero, ¿qué pasa si tenemos un repositorio local y queremos establecerlo en remoto? Source Tree también nos ofrece la posibilidad de configurarlo para este fin.

Accedemos a la ventana inicial de Source Tree y vamos a la pestaña local:

Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

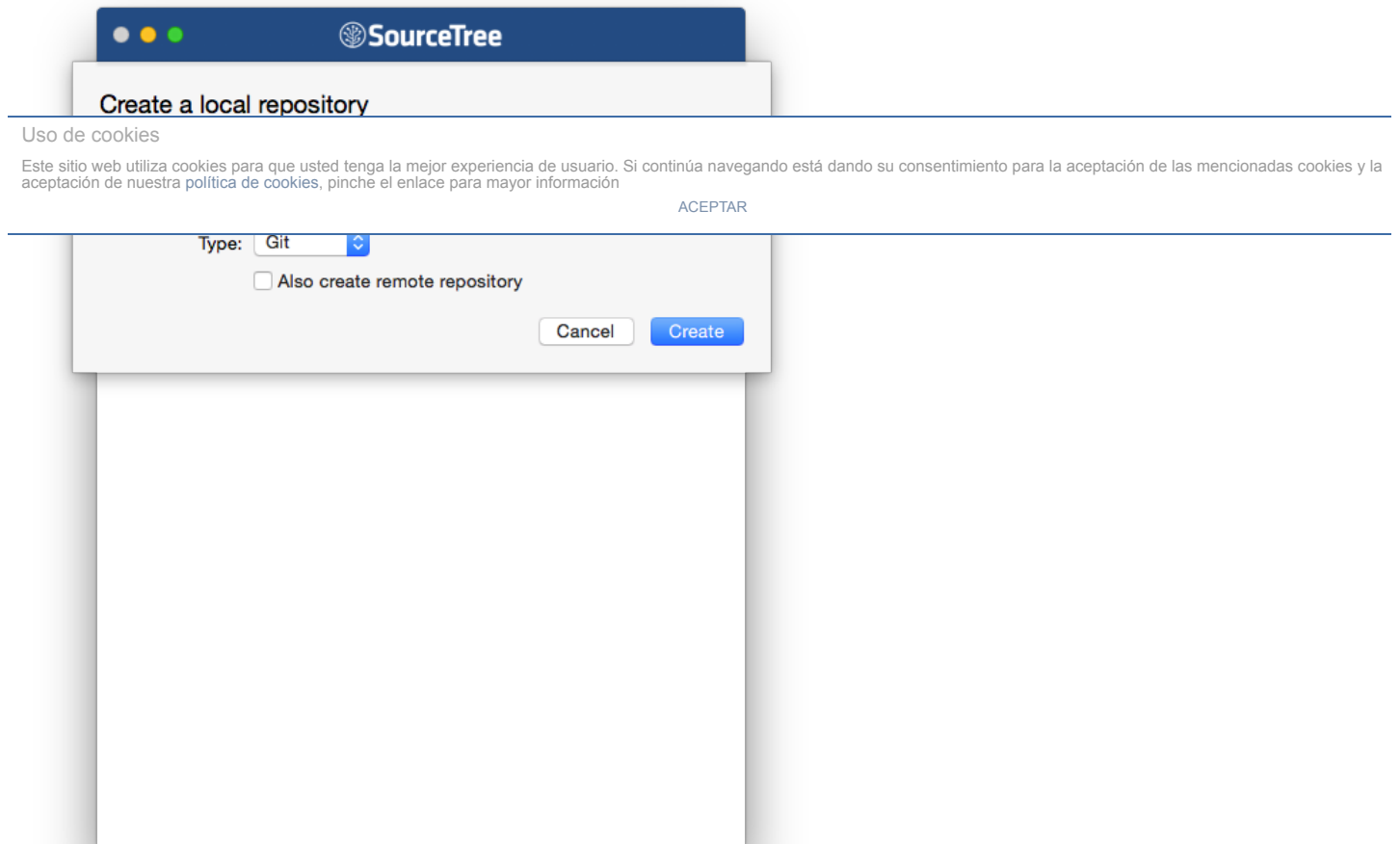
ACEPTAR



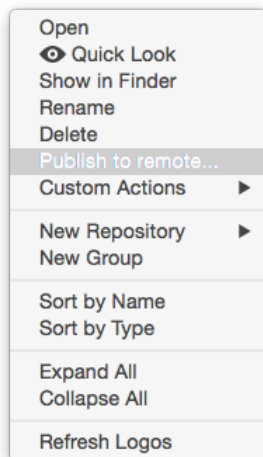
Aquí vemos el proyecto que hemos creado anteriormente, que ya está en local. Ahora vamos a crear un nuevo proyecto. Para ello cogemos la carpeta del proyecto y la arrastramos a la ventana de Source Tree, con la pestaña local seleccionada. Nos saldrá una ventana con:.

- **Destination Path:** ruta del proyecto desde donde lo hemos arrastrado.
- **Name:** el nombre que va a tener el proyecto.
- **Type:** el tipo de sistema de versionado a utilizar.
- **Checkbox** de creación de repositorio externo a la vez, que no marcaremos.

NOTA: Aunque salga el check de "Also create a remote repository", nosotros crearemos el repositorio en dos pasos.

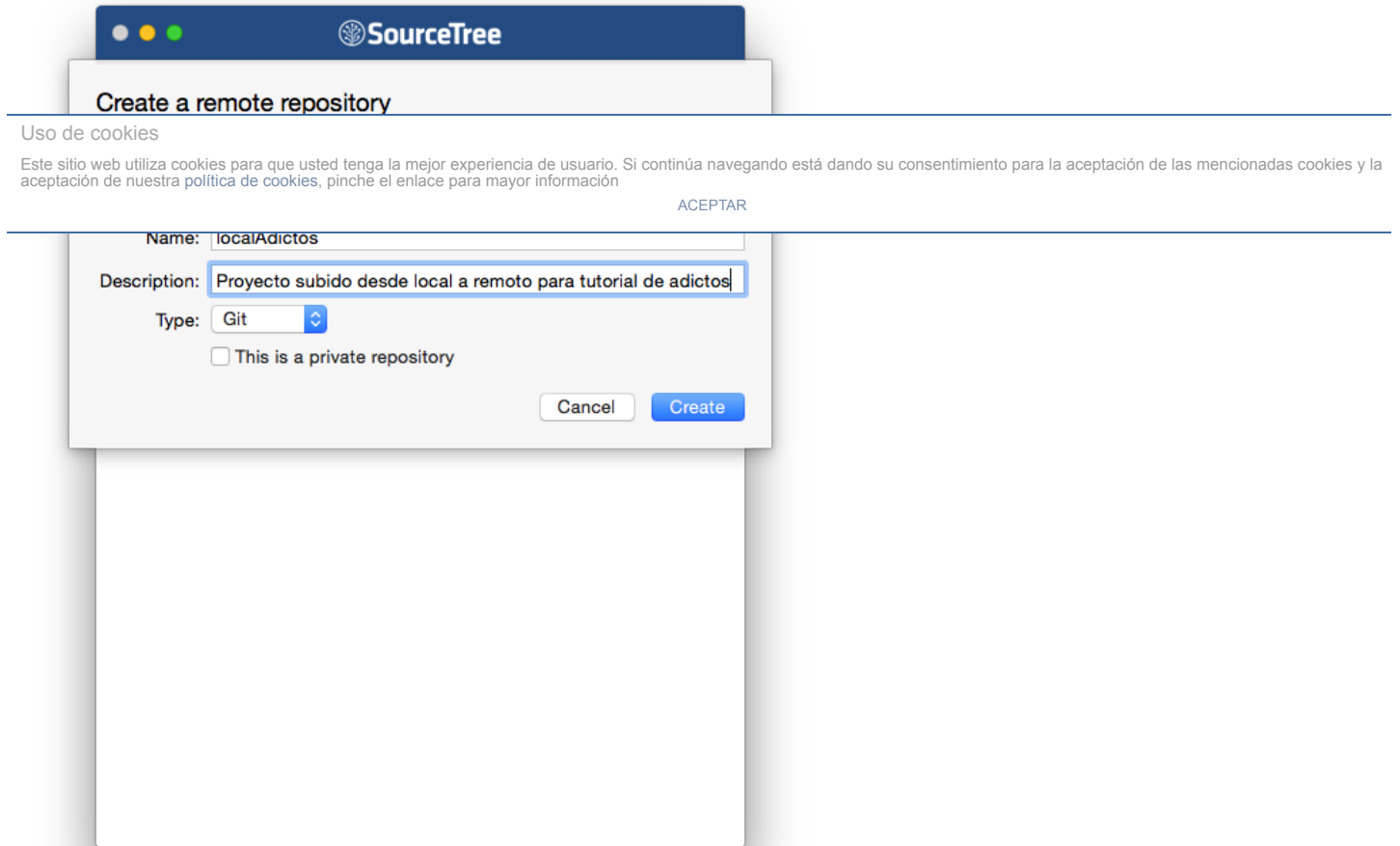


Para que este repositorio sea publicado en remoto, hacemos click derecho sobre él, y seleccionamos *"Publish to remote..."*, dónde deberemos especificar el lugar del repositorio local (por defecto será el lugar desde donde hemos arrastrado la carpeta), el nombre del repositorio y el tipo, que en nuestro caso será git.



En este nuevo desplegable configuramos:

- **Account:** que cuenta de GitHub quieres utilizar para crear el repositorio.
- **Owner:** el propietario de la cuenta.
- **Name:** el nombre que va a tener ese repositorio en remoto (suele llamarse igual).
- **Description:** descripción opcional para definir el proyecto.
- **Type:** tipo de repositorio.
- Un **checkbox** que indica si es un repositorio privado o público. En nuestro caso lo desmarcamos ya que usamos el servicio gratuito de Git, que solo permite la creación de repositorios públicos.



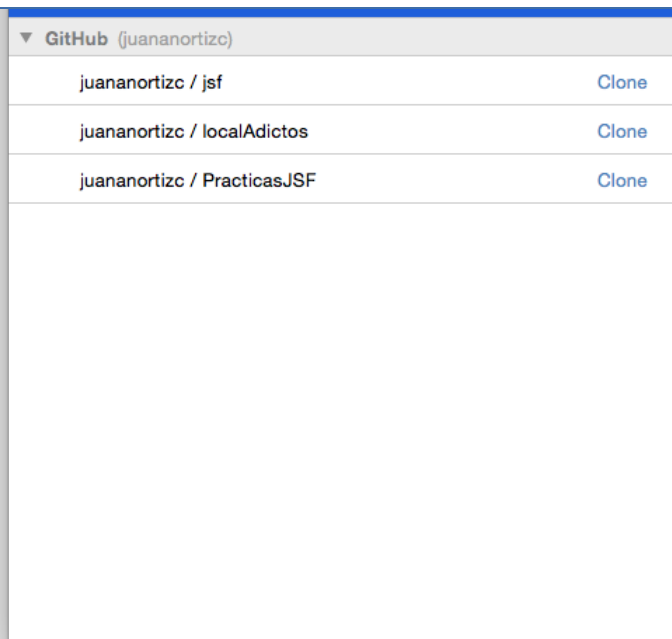
Ahora si accedemos a nuestra pestaña remote en source tree, comprobaremos como el repositorio ha sido creado.



Uso de cookies

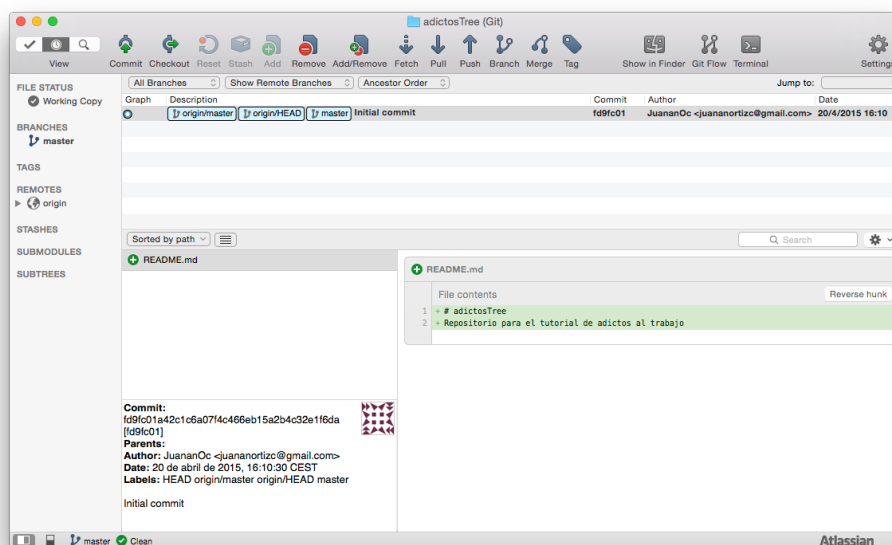
Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR



6. Operaciones básicas de versionado con Source Tree

En los dos pasos anteriores hemos visto las dos formas de crear un repositorio, ya sea clonando uno existente, o creándolo a partir de un proyecto local para poder trabajar con ellos. Ahora veremos las principales herramientas de Source Tree. Para el ejemplo que estamos manejando actualmente, nos es indiferente usar un proyecto u otro, así que usaremos adictosTree. Empecemos echándole un vistazo a la interfaz:



Comenzando por la barra lateral, echemos un vistazo a los elementos más relevantes:

- **File Status:** Como su nombre indica muestra el estado de los ficheros. Siempre que cambiemos algo en algún fichero, cuando pongamos el foco en la ventana del Source Tree, aquí se verán reflejados los

cambios en los ficheros. Como este es el proyecto que queremos subir de local al remoto, aquí aparecen todos los archivos del proyecto, ya que aunque el repositorio remoto se ha creado, ningún dato ha sido enviado todavía. Esta pestaña se corresponde con el siguiente comando de git:

```
1 git status
```

Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR

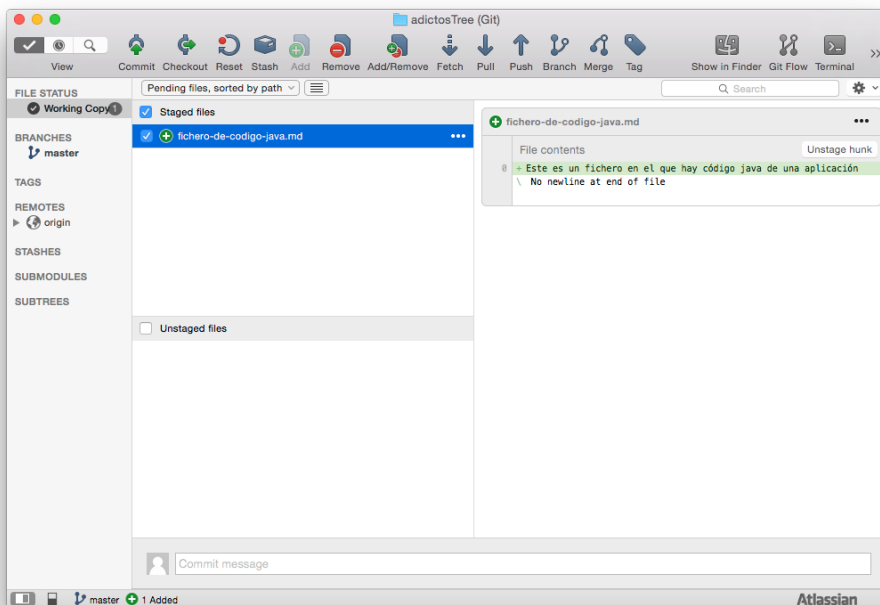
La información de "Branches" y "Remotes" se puede ver en git con el comando:

```
1 git branch -a
```

En la barra superior aparecen las acciones principales que solemos hacer con git: Commit, Checkout, Reset, Stash, Remove, Add/Remove, Fetch, Pull, Push Branch, Merge y Tag

Ahora procedemos a realizar un **push** de la aplicación. Previamente, hemos tenido que modificar, crear o eliminar algún fichero previo del proyecto. Una vez hecho, nos vamos a **Working Copy**. Una vez en esta ventana, en la parte inferior aparecen los unstaged files, ficheros que han sido modificados, creados o destruidos, pero que no están en el repositorio local(STAGE). Para añadirlos, simplemente tenemos que pulsar en Add en el menú superior, o en el check encima de todos los ficheros para añadirlos de golpe. Esto en línea de comandos se corresponde con:

```
1 git add [nombre-fichero]
```



Ahora realizamos un **commit**, para añadir estos ficheros al repositorio local, y nos saldrá una ventana en la parte inferior, cuyo contenido es:

- **Commit options:** donde podemos decidir entre distintas opciones del commit, como la de deshacer un commit por defecto.
- **Caja de texto:** en la que se introduce el comentario del commit.
- **Check para push:** para hacer un push inmediatamente después del commit.



Esta pequeña ventana es equivalente al comando:

```
1 git commit -m "Primer commit de la aplicación adictosTree"
```

Si nos vamos al apartado **Branches** del menú lateral, veremos que ha salido la rama principal por defecto master. en esta ventana encontraremos la información más importante a la hora de hacer un seguimiento de las

versiones que vayamos subiendo al repositorio.

- **Graph:** nos muestra de forma visual como se están desarrollando las distintas ramas(lo comprobaremos cuando hagamos una nueva branch)

- **Description:** indica los repositorios que se encuentran en esta versión así como la descripción que se

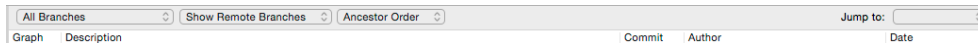
Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR

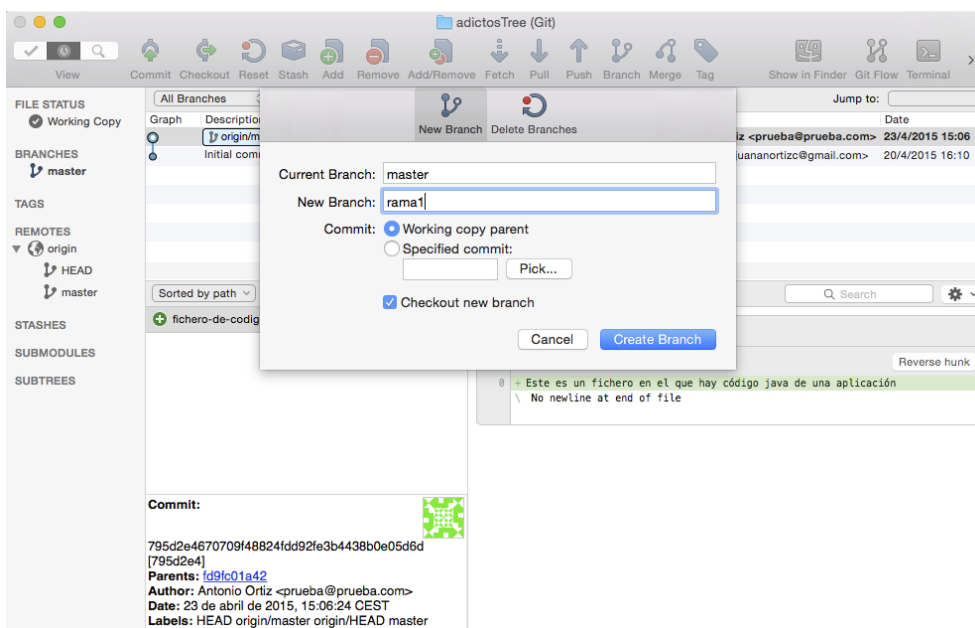
git del repositorio

- **Author:** nos indica el autor del commit, así como su correo electrónico.
- **Date:** indica la hora y la fecha a la que se hizo el commit.



A continuación crearemos una nueva rama, modificaremos un fichero del proyecto tanto en la rama principal como en la rama creada, y haremos un merge de la misma. El objetivo es ver como nos muestra las diferentes ramas el grafo de la aplicación, así como ver como realizar estos dos comando básicos de git.

Para crear una nueva rama de trabajo, basta que pinchar en **Branch** en el menú superior. Esto desplegará la siguiente ventana:



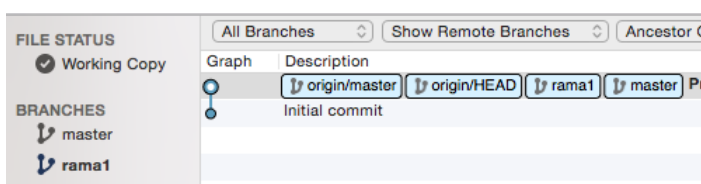
En esta ventana puedes configurar:

- **Current Branch:** indica la rama en la que estamos actualmente.
- **New Branch:** indica el nombre que va a tener la nueva rama.
- **Commit:** Un checkbox que indica desde que commit quieres realizar la creación de la rama nueva(por defecto desde donde se encuentre la rama actual).
- **Checkbox** que indica si quieres realizar un checkout de la rama creada.

Tal y como hemos creado rama 1 se corresponde a los comandos de git de:

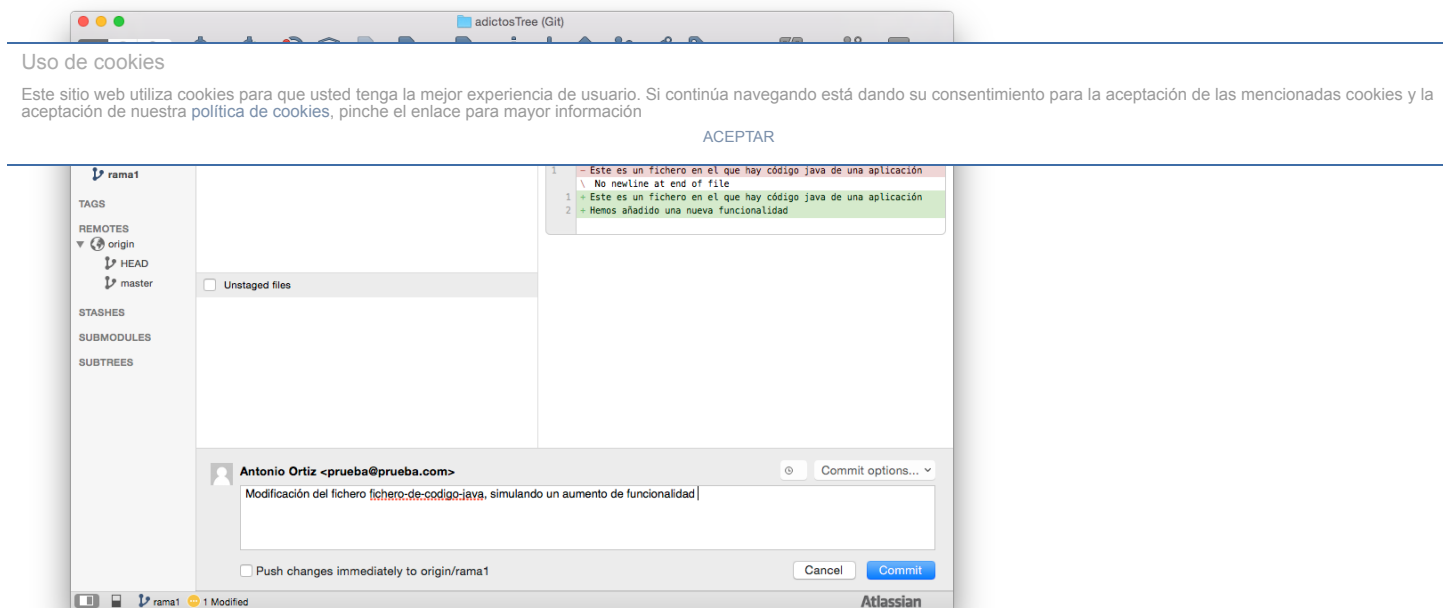
```
1 git checkout -b rama1
```

Tras pulsar el botón "Create Branch", observamos como en el menú de la izquierda, en Branches se nos ha creado nuestra nueva rama. También podemos observar como en la descripción aparece esta nueva rama en el commit actual, que es el que hemos elegido.

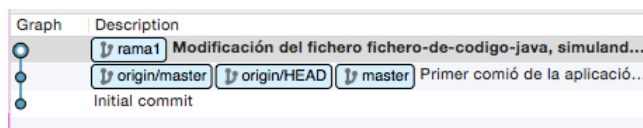


Ahora procedemos a realizar cambios en esta rama, para luego realizar un **merge**(comprobaremos los cambios realizados en el menu izquierdo, en working copy)

Una vez modificado el archivo o creado uno nuevo, vamos a nuestro working copy y realizamos un commit en la rama creada *rama1*.



Comprobamos el estado actual de las ramas en el graph:

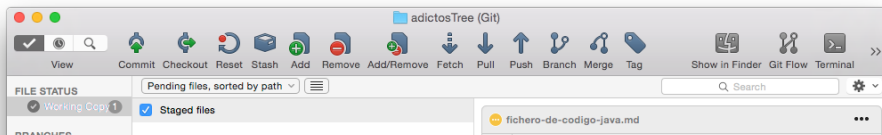


Antes de realizar el merge, vamos a modificar el fichero desde la rama principal, para que veamos como se ven reflejados estos cambios en el grafo que nos muestra a aplicación. Para ello, hacemos un cambio de rama, a la rama principal (master) simplemente haciendo doble click en la rama del menú de la derecha (comprobamos como ahora el nombre en negrita es master).

Esto es correspondiente al siguiente comando de git:



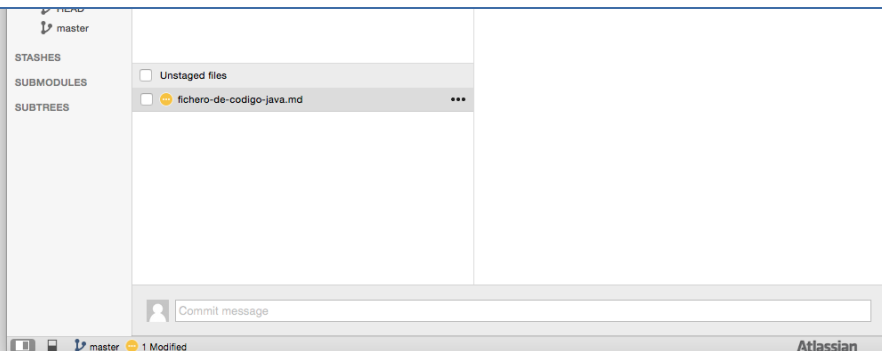
Modificamos el fichero desde la rama principal y hacemos un commit (comprobar como el contenido del fichero a la derecha no se corresponde con el contenido que tenía el fichero al hacer el commit en la rama1):



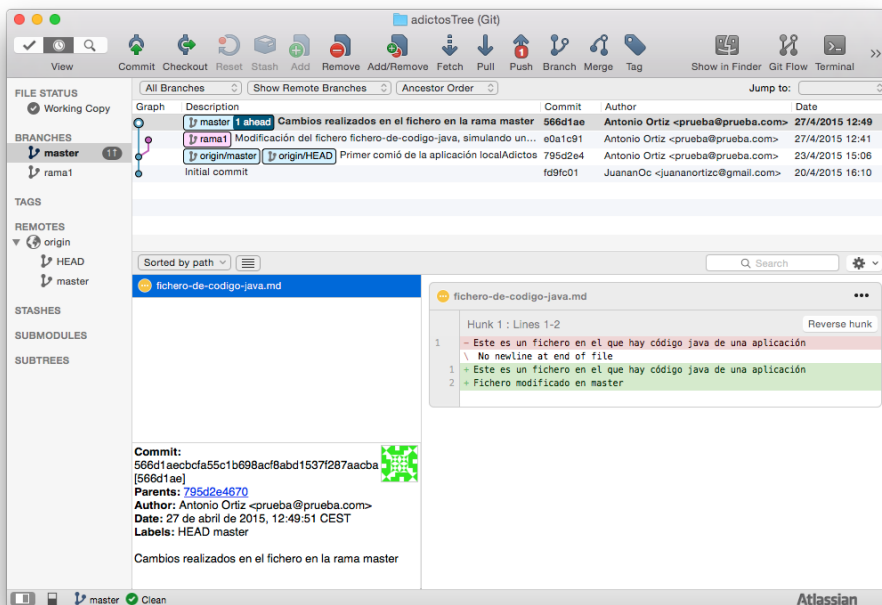
Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR

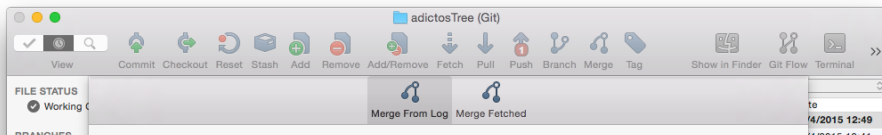


Si ahora observamos el grafo, podemos comprobar como nos muestra las distintas ramas que están siendo creadas. Observar también como al modificar el fichero en la rama principal, este se encuentra una versión por delante del que se encuentra en remoto (1 ahead):



Gracias a la herramienta gráfica podemos ver de forma visual las ramas en las que estamos trabajando, así como los cambios realizados en cada una de ellas leyendo los commits en la parte inferior.

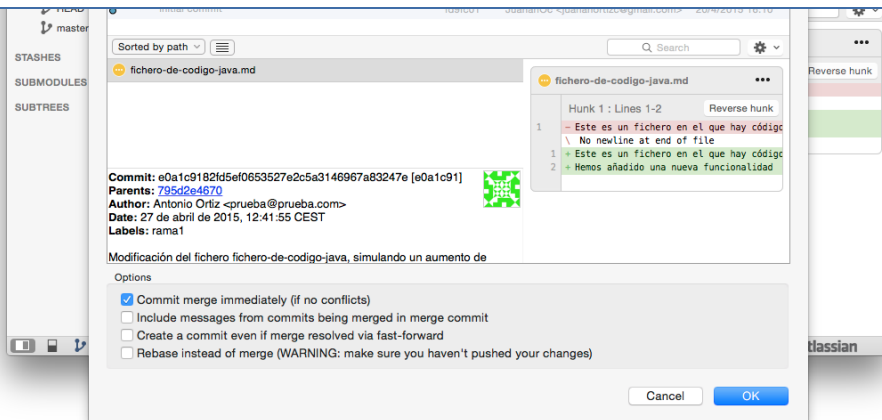
Para finalizar, realizaremos un **merge** de rama1 a master, y subiremos los cambios al servidor remoto(recordad que toda la creación de las ramas ha sido en local, ya que no hemos hecho ni un solo push). Para realizar un merge, basta con pulsar el botón merge en el menú superior.



Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR



En esta ventana observamos como tenemos que elegir una rama para que se una a nuestra rama actual(nosotros estamos actualmente en master). Por lo que solo tenemos que seleccionar la rama en el grafo. Dejamos la opción por defecto de que realice un commit si no han surgido conflictos, y aceptamos.

Como era de esperar, nos ha surgido un conflicto debido a la modificación del mismo fichero en dos ramas diferentes, así que saldrá un mensaje de aviso como el siguiente:



Para resolverlo, lo más común es ir al archivo en conflicto, y resolver el error manualmente, aunque si tienes claro que versión del archivo quieres dejar, en working copy, hacer click derecho en el archivo, y tienes una opción de resolución de conflictos, que te da las opciones tanto de conservar el fichero de tu rama, de la rama ajena, o abrir un editor externo para resolver el conflicto.

Open
Show In Finder
Copy Path To Clipboard
Open In Terminal
Quick Look

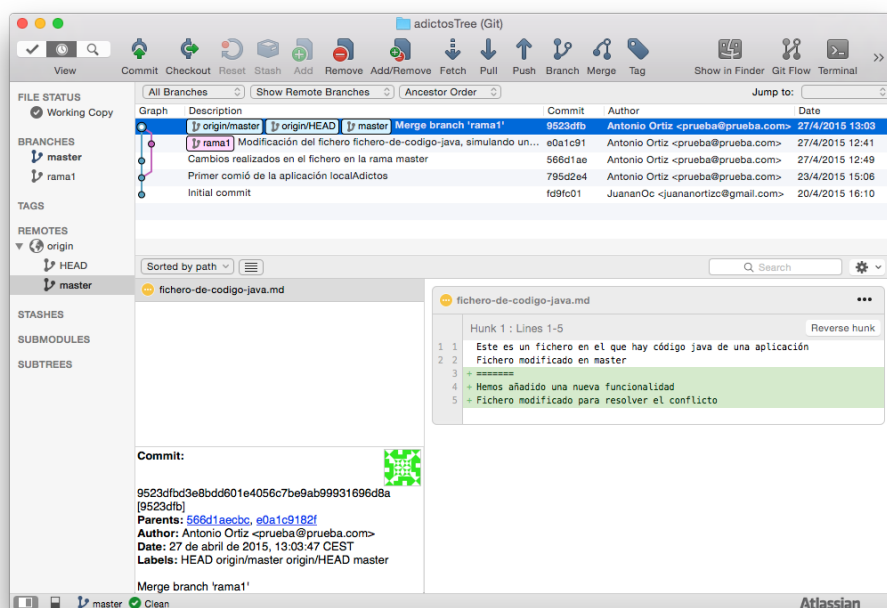
Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR

Add to index
Unstage from index
Remove
Stop Tracking
Ignore...
Commit Selected...
Reset...
Reset to Commit...
Resolve Conflicts
Custom Actions
Log Selected...
Blame Selected...
Copy...
Move...
Expand All
Collapse All

En nuestro caso resolvemos el error manualmente, y hacemos un commit y un push de la aplicación, como habíamos visto previamente, siendo el resultado:



Para finalizar, si ya hemos terminado con la rama, la eliminamos haciendo click derecho en la rama a eliminar, y seleccionando la opción *Delete [nombre rama]*.

En comandos sería equivalente a:

```
1 git branch -d [nombre-rama]
```

Checkout rama1
Merge rama1 into master
Rebase current changes onto rama1

Push to

Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR

Create Pull Request...

Con esto ya habríamos visto las principales funcionalidades de Source Tree. Tiene muchas más herramientas, ¡te aconsejo encarecidamente que trastees un poco con la interfaz!

7. Conclusiones

En resumen:

- ¡Dile adiós a la línea de comandos!. Usando la aplicación de escritorio puedes manejar todos tus repositorios, tanto locales como remotos a través de una interfaz simple.
- Simplifica el control de versiones para el equipo. Operaciones como create, clone, commit, push, pull y merge solo con par de clicks.
- Es bueno tanto para los que están iniciándose, como para los más expertos.

8. Referencias

- <http://www.sourcetreeapp.com/>
- <http://git-scm.com/docs/user-manual.html>

git

repositorio

Source Tree

Comparte este artículo!



Esta obra está licenciada bajo [licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

Entradas relacionadas

[Cómo liberar/distribuir versiones de proyectos Maven+Java con submódulos Git en un entorno CI](#) (julio 20, 2017)

[Cómo borrar ficheros del histórico de Git, y en general cómo manipular todo el histórico](#) (febrero 2, 2017)

[Problemas con macOS Sierra y el SSH para acceder a repositorios Git](#) (enero 24, 2017)

Respuestas (5)



Jorge

junio 9, 2015 at 9:40 am · Responder

Muy bueno, útil e interesante. Sigue así!



Rubén

junio 9, 2015 at 3:56 pm · Responder

Fantástico tutorial de esta útil aplicación, está muy bien explicado.

**Ainhoa**

junio 9, 2015 at 9:23 pm · Responder

Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR

**Lito**

agosto 18, 2015 at 8:46 am · Responder

Gracias por compartir :D.. básico, pero fundamentos esenciales!!

**Ludwin**

enero 21, 2017 at 12:30 am · Responder

Gracias por la clara descripción y por la ayuda gráfica que hace todo más claro.

Aquí dejo mi aporte, si desean actualizar el estado del “origin”

<https://www.ignieweb.com/es/actualizar-estado-del-origin-en-sourcetree-2/>

Deja un comentario

Name (required)

E-Mail (required)

Website

Enviar comentario

Lo último

Securizar un API REST
utilizando JSON Web Tokens
Introducción a StencilJS y demo
de integración con Angular
Javassist: manipulando el
bytecode de una aplicación Java.
Firebase Cloud Messaging:
mensajería gratuita en la nube
Disminuir tiempos de carga del
Discover en Kibana 5

Datos de contacto

Edificio BestPoint Avd. de Castilla,
1, Planta 2, Oficina 21B (San
Fernando de Henares)
Phone: 916 75 33 06
E-Mail:
adictos@adictosaltrabajo.com
Web: <https://www.autentia.com>

Powered by



Copyright 2003-2016 © All Rights Reserved | Texto legal y condiciones de
uso

Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información

[ACEPTAR](#)
