

# Control de versiones con Subversion



# Uso básico de Subversion

Comandos básicos y forma de trabajo de Subversion



# RTFM

- El comando más importante para empezar a usar Subversion:

**svn help**

- El cliente de la línea de comandos de subversion está auto-documentado.

**svn help comando**

Nos muestra la sintaxis, parámetros, etc. del comando.

# Agregando datos al repositorio

## **svn import**

El comando **svn import** es una forma rápida de copiar una jerarquía de archivos sin versionar a un repositorio.

No necesita una copia local, y los archivos se comitean inmediatamente al repositorio.

Se usa generalmente para comenzar a llevar un control de versiones de un proyecto/directorio/etc.

# Agregando datos al repositorio

## Ejemplo:

```
$ svnadmin create /var/svn/repositorio_nuevo
$ svn import proyecto file:///var/svn/repositorio_nuevo/algun/proyecto -m "Primer Import"
Adding      proyecto/Main.java
Adding      proyecto/Persona.java
Adding      proyecto/Pokemon.java
Adding      proyecto/subdirectorio/OtraClase.java
Adding      proyecto/otrosArchivos

Committed revision 1.
```

**NOTA:** Importar los archivos al repositorio no convierte al árbol original de directorios y archivos en una copia local. Los archivos originales no quedan bajo control de versiones. Es necesario hacer un checkout para obtener una copia local.

# Disposición recomendada repositorio Subversion

La disposición sugerida por Subversion para la organización de un repositorio es la siguiente:

**trunk** - Línea principal de desarrollo.

**branches** - Directorio para albergar ramas alternativas de desarrollo.

**tags** - Directorio con versiones etiquetadas.

# Checkout inicial

- Para empezar a usar un repositorio Subversion, generalmente se empieza por hacer un **checkout** de un proyecto.
- Esto crea una copia local (**working copy**) del proyecto.
- Esta copia corresponde a la versión **HEAD** (revisión más reciente) del proyecto en el repositorio.

# Checkout inicial

▪ Nuestra **working copy** es como cualquier colección de archivos y directorios de nuestro sistema. Podemos comenzar a cambiarlos inmediatamente.

▪ Subversion detecta los cambios hechos sobre los archivos, pero debemos notificarle de las demás acciones.

Por ejemplo si queremos mover, o copiar archivos/directorios, debemos usar los comandos de Subversion en vez de los de nuestro sistema.



# Ciclo común de trabajo

**1.Actualizar tu copia local:** Comandos más comunes durante el proceso de desarrollo con Subversion  
`svn update`

**2.Hacer cambios:**

`svn add`  
`svn delete`  
`svn copy`  
`svn move`

**3.Examinar tus cambios:**

`svn status`  
`svn diff`

**4.Posiblemente revertir algunos cambios**

`svn revert`

**5.Resolver conflictos (merge con los cambios de otros)**

`svn update`  
`svn resolve`

**6.Commit de tus cambios:**

`svn commit -m "Mensaje"`

# Actualizar la copia local - update

- Si trabajamos en equipo, queremos obtener cualquier cambio que otros hayan realizado antes de empezar a programar:

**svn update**

- Si alguno de nuestros archivos fue modificado y "commiteado" en el servidor, nuestra versión se actualizará a la versión HEAD.

- A leer a la consola: **svn help update**

# Trabajando con el working copy

*algo* = archivo, directorio o enlace simbólico

## **svn add *algo***

Listar *algo* para ser agregado al repositorio. Si *algo* es un directorio, se agrega todo lo que esté debajo de *algo*. Si solo se quiere agregar el directorio *algo* sin lo que contiene, usar `–depth`

## **svn delete *algo***

Listar *algo* para ser borrado del repositorio. Si es un archivo o enlace, se borra de la copia local inmediatamente. Si es un directorio se marca para ser borrado.

# Trabajando con el working copy

## **svn copy algo nada**

Crear un nuevo ítem nada como duplicado de algo y listarlo para agregarlo. SVN registra que nada proviene de algo.

## **svn move algo nada**

Este comando hace lo mismo que si hiciéramos svn copy algo nada; svn delete algo. Nada se lista para ser agregado como copia de algo, y algo se lista para remover.

## **svn mkdir todo**

Este comando hace lo mismo que si hiciéramos mkdir todo; svn add todo. Se crea el directorio todo y se lista para agregar.

# Revisar los cambios

Al terminar de aplicar cambios al repositorio antes de hacer un commit es buena práctica revisar qué se cambió:

- Esto ayuda a hacer un mensaje de log más preciso.
- Se descubren cambios imprevistos (podemos revertirlos)
- Oportunidad de revisar y examinar los cambios antes de publicarlos.

**svn status** - ver los cambios

**svn diff** - detalles de los cambios

# svn status

svn status detecta los cambios hechos a los directorios y archivos.  
Los códigos de status que retorna son:

## **A algo**

algo está listado para agregar al repositorio.

## **C algo**

algo está en conflicto, los cambios recibidos del servidor en un update se solapan con cambios locales y no fueron resueltos en el update. Hay que resolver este conflicto antes de hacer un commit.

## **D algo**

algo está listado para ser borrado del repositorio

## **M algo**

El contenido de algo ha sido modificado.

# svn status

- Se puede conocer el estado de un archivo en particular pasándolo como parámetro:

```
svn status algo  
A      algo
```

- El parámetro **-v** (**--verbose**) muestra el estado de todo lo que haya en la copia local
- El parámetro **--show-updates** (**u**) contacta con el repositorio y agrega información sobre lo que está desactualizado en nuestra copia local.

# svn commit

Comando para publicar los cambios en el repositorio.

**Debe recibir un mensaje** para adjuntar a la revisión creada al momento de hacer el commit.

Como parámetro con `-m` (`--message`):

```
svn commit -m "Estoy comiteando algo"
```

En archivo de texto:

```
svn -file (-F) mensaje.txt
```

En caso de no especificar un mensaje, subversion ejecuta un editor de texto automáticamente para escribir un mensaje.



# Otras herramientas

## **svn diff**

Correr este comando sin argumentos muestra los cambios con el repositorio en formato diff unificado.

Las líneas removidas se muestran con -

Las líneas agregadas se muestran con +

## **svn revert**

Permite deshacer los cambios hechos en la copia local. También se puede deshacer cualquier operación listada como "agregar", "borrar", para volver al estado anterior.

# Esta presentación es libre

Copyright © 2008 – Fernando Briano - <http://picandocodigo.net>

Esta presentación es un trabajo derivado de:

**Version Control with Subversion** - Copyright © 2002-2008 Ben Collins-Sussman, Brian W. Fitzpatrick, C. Michael Pilato.

<http://svnbook.red-bean.com>

Licencia del trabajo original:

**Creative Commons Attribution License**

<http://creativecommons.org/licenses/by/2.0/>

Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

Este trabajo se publica bajo la siguiente licencia:

**Creative Commons Attribution License**

<http://creativecommons.org/licenses/by/3.0/>

**Usted es libre de:**

- copiar, distribuir y comunicar públicamente la obra
- hacer obras derivadas

**Bajo las condiciones siguientes:**

- Reconocimiento. Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciador (pero no de una manera que sugiera que tiene su apoyo o apoyan el uso que hace de su obra).
- Al reutilizar o distribuir la obra, tiene que dejar bien claro los términos de la licencia de esta obra.
- Alguna de estas condiciones puede no aplicarse si se obtiene el permiso del titular de los derechos de autor
- Nada en esta licencia menoscaba o restringe los derechos morales del autor.