Búsqueda y reemplazo recursivo con sed en directorios

Contexto

Estrategias para localizar y modificar cadenas de texto en múltiples archivos de forma recursiva, con capacidad de filtrar por tipo de archivo y excluir líneas comentadas.

Escenario típico

- Necesidad de cambiar nombres de variables, funciones o constantes en toda una base de código
- Refactorización de nombres de configuración
- Migración de valores legacy a nuevas nomenclaturas
- Requiere precisión para no modificar comentarios ni código inactivo

Estrategias de búsqueda

1. Localización básica (análisis previo)

```
# Buscar en todos los archivos
find . -type f -exec grep -l "PATRON_BUSQUEDA" {} \;

# Con contexto (3 líneas antes y después)
find . -type f -exec grep -n -B3 -A3 "PATRON_BUSQUEDA" {} + | less

# Múltiples patrones (OR lógico)
find . -type f -exec grep -l "PATRON_1\|PATRON_2\|PATRON_3" {} \;
```

2. Filtrado por tipo de archivo

```
# Solo archivos con extensión específica
find . -type f -name "*.php" -exec grep -l "PATRON" {} \;
find . -type f -name "*.js" -exec grep -l "PATRON" {} \;
find . -type f -name "*.py" -exec grep -l "PATRON" {} \;

# Múltiples extensiones
find . -type f \( -name "*.php" -o -name "*.inc" \) -exec grep -l "PATRON" {} \;

# Excluir directorios específicos
find . -type f -name "*.php" -not -path "*/vendor/*" -not -path "*/node_modules/*" \
  -exec grep -l "PATRON" {} \;
```

3. Análisis con numeración de líneas

```
# Ver archivo:línea:contenido
find . -type f -name "*.php" -exec grep -Hn "PATRON" {} \;

# Con AWK para formato personalizado
find . -type f -name "*.php" -exec awk '/PATRON/ {print FILENAME ":" NR ":" $0}' {} +
```

Estrategias de reemplazo

1. Reemplazo básico (SIN backup automático)

```
# Reemplazar en todos los archivos de un tipo
find . -type f -name "*.php" -exec sed -i 's/PATRON_VIEJO/PATRON_NUEVO/g' {} \;

# Múltiples reemplazos en una pasada
find . -type f -name "*.php" -exec sed -i '
  s/PATRON_1/NUEVO_1/g
  s/PATRON_2/NUEVO_2/g
  s/PATRON_3/NUEVO_3/g
' {} \;
```

Nota: `-i` modifica archivos en sitio. En BSD/macOS usar `sed -i ''`

2. Reemplazo CON backup automático

```
# Crea archivo.php.bak antes de modificar
find . -type f -name "*.php" -exec sed -i.bak 's/PATRON_VIEJO/PATRON_NUEVO/g' {} \;

# Limpiar backups después de verificar
find . -type f -name "*.php.bak" -delete
```

3. Excluir líneas comentadas

PHP/JavaScript/C-style

```
# Excluir líneas que empiezan por #, //, /* o *
find . -type f -name "*.php" -exec sed -i.bak '/^\s*[#*\/]/! s/PATRON/NUEVO/g' {} \;
```

Desglose del patrón:

- `/^\s*[#*\/]/!` → Negación: aplica solo si NO cumple
- `^\s*` → Inicio de línea con posibles espacios/tabs
- `[#*\/]` → Caracteres de comentario: `#`, `*`, `/`
- `!` → Negación lógica

Versión robusta (múltiples tipos de comentario)

```
find . -type f -name "*.php" -exec sed -i.bak '
/^\s*#/! {
  /^\s*\/\//! {
    /^\s*\/\*/! {
      /^\s*\*/! {
        s/PATRON_1/NUEVO_1/g
        s/PATRON_2/NUEVO_2/g
      }
    }
  }
}
' {} \;
```

Esto excluye:

- `# comentario estilo shell/Python`
- `// comentario estilo C++/JavaScript`
- `/* inicio de bloque comentario */`
- `* líneas dentro de bloque comentario`

Python

```
# Excluir líneas que empiezan por #
find . -type f -name "*.py" -exec sed -i.bak '/^\s*#/! s/PATRON/NUEVO/g' {} \;
```

Shell scripts

```
# Excluir líneas que empiezan por #
find . -type f -name "*.sh" -exec sed -i.bak '/^\s*#/! s/PATRON/NUEVO/g' {} \;
```

4. Alternativa con Perl (más flexible)

```
# Perl permite regex más complejas
find . -type f -name "*.php" -exec perl -i.bak -pe '
  s/PATRON_VIEJO/PATRON_NUEVO/g unless /^\s*[#*\/]/
' {} \;

# Múltiples patrones con hash de sustituciones
find . -type f -name "*.php" -exec perl -i.bak -pe '
  BEGIN {
    %replace = (
      "PATRON_1" => "NUEVO_1",
      "PATRON_2" => "NUEVO_2",
      "PATRON_3" => "NUEVO_3"
    );
  }
  unless (/^\s*[#*\/]/) {
    s/$_/$replace{$_}/g for keys %replace;
  }
' {} \;
```

Flujo de trabajo recomendado

```
# PASO 1: Análisis inicial (¿qué se va a cambiar?)
find . -type f -name "*.php" -exec grep -Hn "PATRON_VIEJO" {} \; | less

# PASO 2: Análisis excluyendo comentarios
find . -type f -name "*.php" -exec awk '
```

```
  !/^\s*[#*\/]/ && /PATRON_VIEJO/ {
    print FILENAME ":" NR ":" $0
  }
' {} + | less

# PASO 3: Prueba en un solo archivo
sed -i.test '/^\s*[#*\/]/! s/PATRON_VIEJO/PATRON_NUEVO/g' ruta/archivo_prueba.php
diff ruta/archivo_prueba.php ruta/archivo_prueba.php.test

# PASO 4: Aplicar con backup (EJECUTAR)
find . -type f -name "*.php" -exec sed -i.bak '/^\s*[#*\/]/! s/PATRON_VIEJO/PATRON_NUEVO/g' {}
\;

# PASO 5: Revisar cambios (si hay control de versiones)
git diff
git status

# PASO 6: Validar funcionamiento
# [Ejecutar tests, verificar aplicación]

# PASO 7: Limpiar backups si todo OK
find . -name "*.php.bak" -delete
```

Casos de uso comunes

Cambio de prefijo de base de datos

```
# MySQL/MariaDB: cambiar prefijo de tablas en dumps
find . -type f -name "*.sql" -exec sed -i.bak 's/prefijo_viejo_/prefijo_nuevo_/g' {} \;
```

Refactorización de nombres de clase

```
# PHP: cambiar nombre de clase
find . -type f -name "*.php" -exec sed -i.bak '/^\s*[#*\/]/! {
  s/class OldClassName/class NewClassName/g
  s/OldClassName::/NewClassName::/g
  s/new OldClassName/new NewClassName/g
```

```
}' {} \;
```

Migración de configuración

```
# Cambiar nombres de constantes de configuración
find . -type f \( -name "*.php" -o -name "*.inc" \) -exec sed -i.bak '/^\s*[#*\/]/! {
  s/OLD_CONFIG_NAME/NEW_CONFIG_NAME/g
  s/OLD_DB_HOST/NEW_DB_HOST/g
  s/OLD_API_KEY/NEW_API_KEY/g
}' {} \;
```

Actualización de rutas

```
# Cambiar rutas legacy
find . -type f -name "*.php" -exec sed -i.bak '/^\s*[#*\/]/! {
  s|/var/www/old_path/|/var/www/new_path/|g
  s|old-domain\.com|new-domain.com|g
}' {} \;
```

Limitaciones conocidas

1. Comentarios multilínea

El método basado en `^\s*` solo detecta comentarios al **inicio de línea**. No detecta:

```
$var = "value"; /* comentario inline */ $otra = "PATRON"; // no se excluirá
```

Para estos casos, considerar un parser específico del lenguaje.

2. Cadenas literales

Si `PATRON` aparece dentro de strings, será reemplazado:

```
$sql = "SELECT * FROM old_table"; // se cambiará "old_table"
```

Para evitarlo se requiere análisis sintáctico completo.

3. Bloques multilínea

Comentarios que abarcan múltiples líneas:

```
/*
 * Este es un comentario
 * con PATRON_VIEJO que
 * no será detectado en líneas 2-3
 */
```

Solo la primera línea (`/*`) se excluirá.

Alternativas avanzadas

Ripgrep (rg) + sd

```
# Búsqueda ultrarrápida
rg "PATRON" --type php

# Reemplazo con sd (más intuitivo que sed)
find . -name "*.php" -exec sd "PATRON_VIEJO" "PATRON_NUEVO" {} \;
```

Usando AG (The Silver Searcher)

```
# Listar archivos
ag -l "PATRON" --php

# Con contexto
ag "PATRON" --php -C 3
```

Scripts dedicados por lenguaje

Para refactorizaciones complejas, usar herramientas especializadas:

- **PHP**: PHP-CS-Fixer, Rector
- **JavaScript**: jscodeshift, lebab
- **Python**: rope, bowler

Verificación post-cambio

```
# Contar ocurrencias antes y después
echo "Antes:"
find . -name "*.php" -exec grep -o "PATRON_VIEJO" {} \; | wc -l

echo "Después:"
find . -name "*.php" -exec grep -o "PATRON_NUEVO" {} \; | wc -l

# Verificar que no quedan ocurrencias del patrón viejo
find . -name "*.php" -exec grep -H "PATRON_VIEJO" {} \;

# Si hay control de versiones
git diff --stat
git diff | grep -E "^\+.*PATRON_NUEVO" | wc -l
```

Consejos de seguridad

1. **Siempre hacer backup** antes de modificaciones masivas
2. **Usar control de versiones** (git commit antes de ejecutar)
3. **Probar en un archivo** antes de aplicar a todo el proyecto
4. **Revisar diff completo** antes de commit final
5. **Ejecutar suite de tests** después de cambios
6. **Considerar crear rama temporal** para refactorizaciones grandes

Troubleshooting

"Device or resource busy"

```
# Puede ocurrir con archivos abiertos. Solución: cerrar editores
lsof | grep nombre_archivo
```

"Permission denied"

```
# Verificar permisos
find . -type f -name "*.php" ! -writable
```

```
# Corregir si es necesario
find . -type f -name "*.php" -exec chmod u+w {} \;
```

Sed no modifica archivos

```
# Verificar sintaxis específica del sistema
sed --version  # GNU sed
sed -i.bak ...  # funciona en GNU

# En macOS/BSD
sed -i '' ...   # requiere argumento vacío
```

Última actualización: Octubre 2025
Compatibilidad: Linux (GNU sed), *BSD/macOS (requiere ajustes en `-i`)
Herramientas: find, sed, grep, awk, perl

Revision #2
Created 2025-10-07 05:11:02 UTC by Abkrim
Updated 2025-10-07 05:35:33 UTC by Abkrim