

Como saber que reparar en nuestro rama cuando hago un rebase y hay conflicto

```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
```

```
.git/rebase-apply/patch:70: trailing whitespace.
    <li><a href="">Link 3</a></li>
.git/rebase-apply/patch:83: trailing whitespace.
    Lorem Ipsum is simply dummy text of the printing and typesetting industry.<br>
.git/rebase-apply/patch:85: trailing whitespace.
</p>
warning: 5 lines add whitespace errors.
Falling back to patching base and 3-way merge...
Auto-merging ejercicios/articulo_de_blog.html
CONFLICT (content): Merge conflict in ejercicios/articulo_de_blog.html
error: Failed to merge in the changes.
Patch failed at 0001 update ejercicio clase 13
hint: Use 'git am --show-current-patch' to see the failed patch
Resolve all conflicts manually, mark them as resolved with
"git add/rm <conflicted_files>", then run "git rebase --continue".
You can instead skip this commit: run "git rebase --skip".
To abort and get back to the state before "git rebase", run "git rebase --abort".
(base) mjsorribas@Maximilianos-MacBook-Pro frontendmayo %
```

Miramos que archivos es el conflictivo (la consola nos dice CONFLICT (content): Merge conflict in ejercicios/articulo_de_blog.html). Abrimos el archivo con conflicto

[illegible]

Tratamos de revisar el cambio .. y vemos que solo agrega unas lianas (HEAD) Eliminamos en este ejemplo la linea que dice HEAD (linea14), la separación (linea17) y el incoming change.

```
articulo_de_blog.html
ejercicios > articulo_de_blog.html > html > head > style > article
10 Cada elemento h1 a h6 debe tener una clase css con al menos 2 propiedades y
11 Para los elementos section y article, cada uno debe contener 2 propiedades
12 a una distancia clara de de entre 20 y 40 px.
13 Cada elemento que tenga contenido en parrafos debe contener almenos 2 propi
14 /*corregir tamaño de imagen a un maximo de 250px de alto. remarcar el conte
15 /* Agregar color de fondo por seccion y de documento*/
16 /* Aqui tienen unos ejemplos de otras clases de referencia.*/
17 /* Estructura */
18 article {
19     float: left;
20     width: 70%;
21 }
22 aside {
23     float: right;
24     width: 30%;
25 }
26 /* ----fin Estructura --- */

PROBLEMS 2 OUTPUT TERMINAL DEBUG CONSOLE 1: zsh
hint: Use 'git am --show-current-patch' to see the failed patch
Resolve all conflicts manually, mark them as resolved with
"git add/rm <conflicted_files>", then run "git rebase --continue".
You can instead skip this commit: run "git rebase --skip".
To abort and get back to the state before "git rebase", run "git rebase --abort".
(base) mjsorribas@Maximilianos-MacBook-Pro frontendmayo %
```

Luego de limpiar nuestro archivo. Nos quedará algo así:

Ahora que quedaron los conflictos resueltos guardamos el archivo.

Ejecutamos git add carpeta/nombre_demi_archivo

Ejecutamos git rebase --continue

En el caso de que de este error:

```
(base) mjsorribas@Maximilianos-MacBook-Pro frontendmayo % git add ejercicios/articulo_de_blog.html
(base) mjsorribas@Maximilianos-MacBook-Pro frontendmayo % git rebase --continue
Applying: update ejercicio clase 13
No changes - did you forget to use 'git add'?
If there is nothing left to stage, chances are that something else
already introduced the same changes; you might want to skip this patch.
Resolve all conflicts manually, mark them as resolved with
"git add/rm <conflicted_files>", then run "git rebase --continue".
You can instead skip this commit: run "git rebase --skip".
To abort and get back to the state before "git rebase", run "git rebase --abort".
```

Lo que hacemos es aplicar lo que nos pide que es un git rebase --skip.

```
(base) mjsorribas@Maximilianos-MacBook-Pro frontendmayo % git rebase --skip
Applying: consigna
```

Si por un camino u otro llegaron finalizar el rebase

Ejecutamos git push --force

Empezamos a escribir nuestro código.