Una sugerencia práctica de versionado semático

20220506T1623

Una sugerencia práctica de numeros de versión semánticos: X.Y.p.c

(nota: está basada en el estandar https://semver.org/lang/es/, solo que usando la parte de “PARCHE” para desarrollo/pruebas)

Cara a los usuarios finales, solo existirán las dos primeras cifras X.Y

=> X es versión mayor: cambios no retrocompatibles

=> Y es versión menor: cambios compatibles con cualquier otra con el mismo X

La primera versión X que se lance será la X.1,
luego irá incrementandose X.2, X.3, X.4, etc.

(si decidimos dejar visibles las cuatro cifras,
 lo que el usuario verá sería X.1.0.0, X.2.0.0, X.3.0.0, X.4.0.0, etc.)

Cara al desarrollo, existirán cuatro cifras: X.Y.p.c

=> p.0 es una versión "para que alguien interno la pruebe",
una versión que solo verán los usuarios internos de pruebas: X.Y.1.0, X.Y.2.0, X.Y.3.0, etc. (nota: se pueden abreviar a X.Y.1, X.Y.2, X.Y.3, etc.)

=> p.c es una versión "de compilacion",
una versión que solo verán los programadores: X.Y.p.1, X.Y.p.2, etc.

Por ejemplo:

Si tenemos lanzada a producción (a usuarios finales) la versión 5.7 (5.7.0.0)

Estaremos trabajando internamente en la 5.7.p.c

5.7.0.1,... (*)
hasta que decidamos lanzar a pruebas 5.7.1 (5.7.1.0)

5.7.1.1,... (*)
hasta que decidamos lanzar a pruebas 5.7.2 (5.7.2.0)

5.7.2.1,... (*)
hasta que decidamos lanzar a pruebas 5.7.3 (5.7.3.0)

etc.

Hasta que decidamos lanzar a producción (a usuarios finales) la 5.8 (5.8.0.0)

 (o, si hemos introducido alguna incompatibilidad, la 6.1 (6.1.0.0)

Luego seguiremos trabajando internamente en la 5.8.p.c

5.8.0.1,... (*)
hasta que decidamos lanzar a pruebas 5.8.1 (5.8.1.0)

5.8.1.1,... (*)
hasta que decidamos lanzar a pruebas 5.8.2 (5.8.2.0)

5.8.2.1,... (*)
hasta que decidamos lanzar a pruebas 5.8.3 (5.8.3.0)

O si habiamos lanzado la 6.1 (6.1.0.0) en lugar de la 5.8,
seguriamos trabajando internamente en la 6.1.p.c

6.1.0.1,...
hasta que decidieramos lanzar a pruebas 6.1.1 (6.1.1.0)

6.1.1.1,...
hasta que decidieramos lanzar a pruebas 6.1.2 (6.1.2.0)

6.1.2.1,...
hasta que decidieramos lanzar a pruebas 6.1.3 (6.1.3.0)

Hasta que decidamos lanzar a producción (a usuarios finales) la 5.9 (5.9.0.0)
o la 6.2 (6.2.0.0)

etc.

Resumiendo:

Partiendo de última una versión de producción, por ejemplo 5.8 (5.8.0.0), y de una última versión de pruebas, por ejemplo 5.8.2 (5.8.2.0),

=> en cada **compilación** tras un lanzamiento,
por lo menos poner .c a uno y, si queremos, podemos seguir incrementandolo (*)
5.8.2.1, etc.

=> en cada **lanzamiento de prueba**,
poner .c a cero e incrementar .p:
5.8.3 (5.8.3.0)

=> en cada **lanzamiento de producción**,
poner .d y .c a cero, e incrementar .X e .Y segun corresponda:
5.9 (5.9.0.0) o 6.1 (6.1.0.0)

nota: (*) es opcional incrementar c más allá de uno, ya que es algo solo para la persona o el equipo que está programando.

IMPORTANTE:

Una vez lanzada una versión X.Y (X.Y.0.0) o X.Y.p (X.Y.p.0),

JAMÁS de los jamases SACAR OTRA CON EL MISMO X.Y.p.0

=> en la siguiente compilación:

incrementar .c (poner al menos X.Y.p.1)

=> en el siguiente lanzamiento:

incrementar .p (si es de pruebas) o X.Y (si es de producción)

nota: No pasa nada por lanzar una X.Y para después, en muy poco tiempo, lanzar otra X.Y+1 o incluso un poco más tarde otra X.Y+2 (por ejemplo, si vamos detectando errores graves y tenemos que corregirlos con rapidez). Tampoco pasa nada por llegar hasta versiones del estilo de 57.24.8 o similares...

EL VERSIONADO ES SOLO PARA IDENTIFICAR y saber con qué pieza ÚNICA de software estamos lidiando.

Mejor no mezclar aspectos "estéticos" "marquetinianos" acerca de qué número de versión queda más o menos "bonito" o es más o menos "vendible". La mejor manera de mantener la unicidad es seguir reglas simples.