Rangos de versiones

Índice

1.

Introducción

Introducción

Muchos paquetes NPM siguen la especificación de SemVer pero muchos otros pueden seguir diferentes convenciones o no seguir ninguna.

Introducción

Muchos paquetes NPM siguen la especificación de SemVer pero muchos otros pueden seguir diferentes convenciones o no seguir ninguna.

Por este motivo es necesario el poder especificar en qué rangos de versiones vas a permitir actualizaciones automáticas según el paquete.

Introducción

Muchos paquetes NPM siguen la especificación de SemVer pero muchos otros pueden seguir diferentes convenciones o no seguir ninguna.

Por este motivo es necesario el poder especificar en qué rangos de versiones vas a permitir actualizaciones automáticas según el paquete.

Nota: Aunque SemVer y ComVer dicen que los únicos cambios que pueden romper el API son los MAJOR, es muy recomendable leer el changelog de cada dependencia antes de efectuar un cambio.

2.

Pre-releases versions

Pre-releases

A veces, se quiere poder publicar versiones preliminares para probarlas.

Pre-releases

A veces, se quiere poder publicar versiones preliminares para probarlas.
En estos casos, debemos tagear nuestras versiones de esta forma:

- 0.5.0-alpha1
- 0.5.0-beta1
- 0.5.0-beta2
- 0.5.0-rc1
- 0.5.0-rc2
- 0.5.0

Pre-releases

A veces, se quiere poder publicar versiones preliminares para probarlas.
En estos casos, debemos tagear nuestras versiones de esta forma:

- 0.5.0-alpha1
- 0.5.0-beta1
- 0.5.0-beta2
- 0.5.0-rc1
- 0.5.0-rc2
- 0.5.0

Una release candidate (RC) es una versión muy cercana al release que no va a introducir ninguna nueva funcionalidad. Se conocen como *versiones de refinamiento* antes del final release.

3.

Placeholders

Placeholders

El package.json admite diferentes placeholders que las herramientas como YARN o NPM puede interpretar para saber cuando un paquete debe de ser actualizado.

Placeholders

Para ello, a la hora de especificar la versión podemos hacer uso de los siguientes placeholders:

1. **Signo de intercalación (^1.2.3):** Acepta todos los cambios dentro del MAJOR range. (valid: 1.2.4, 1.2.5, 1.3.0..., invalid: 2.0.0).
2. **Virgulilla (~1.2.3):** Acepta todos los cambios dentro del MINOR range. (valid: 1.2.4, 1.2.5..., invalid: 1.3.0).
3. **Rango (>=1.3.0 <2.0.0):** Acepta cualquier versión dentro del rango.
4. **Versión exacta (1.2.3):** Es el más estricto. Solo admite actualizaciones manuales.
5. **Placeholder X (1.x):** 1.1.1, 1.2.1, etc).

Placeholders

Para ello, a la hora de especificar la versión podemos hacer uso de los siguientes placeholders:

1. **Signo de intercalación (^1.2.3):** Acepta todos los cambios dentro del MAJOR range. (valid: 1.2.4, 1.2.5, 1.3.0..., invalid: 2.0.0).
2. **Virgulilla (~1.2.3):** Acepta todos los cambios dentro del MINOR range. (valid: 1.2.4, 1.2.5..., invalid: 1.3.0).
3. **Rango (>=1.3.0 <2.0.0):** Acepta cualquier versión dentro del rango.
4. **Versión exacta (1.2.3):** Es el más estricto. Solo admite actualizaciones manuales.
5. **Placeholder X (1.x):** 1.1.1, 1.2.1, etc).

Recordar: La especificación de SemVer comienza con la versión 1.0.0. Todo lo anterior a eso no van a funcionar los rangos 1 y 2.

Demo

https://semver.npmjs.com/