Version Class

Use the Version methods to get the version of a first-generation managed package, and to compare package versions.

Namespace

System

Usage

A package version is a number that identifies the set of components uploaded in a package. The version number has the format *majorNumber.minorNumber.patchNumber* (for example, 2.1.3). The major and minor numbers increase to a chosen value during every major release. The *patchNumber* is generated and updated only for a patch release.

A called component can check the version against which the caller was compiled using the `System.requestVersion` method and behave differently depending on the caller's expectations. This allows you to continue to support existing behavior in classes and triggers in previous package versions while continuing to evolve the code.

The value returned by the `System.requestVersion` method is an instance of this class with a two-part version number containing a major and a minor number. Since the `System.requestVersion` method doesn't return a patch number, the patch number in the returned Version object is null.

The `System.Version` class can also hold also a three-part version number that includes a patch number.

Example

This example shows how to use the methods in this class, along with the `requestVersion` method, to determine the managed package version of the code that is calling your package.

```
if (System.requestVersion() == new Version(1,0))
{
    // Do something
}
if ((System.requestVersion().major() == 1)
     && (System.requestVersion().minor() > 0)
     && (System.requestVersion().minor() <=9))
{
    // Do something different for versions 1.1 to 1.9
}
else if (System.requestVersion().compareTo(new Version(2,0)) >= 0)
{
    // Do something completely different for versions 2.0 or greater
}
```

- Version Constructors
- Version Methods

Version Constructors

- **Version(major, minor, patch)**
  Creates a new instance of the `Version` class as a three-part package version using the specified major, minor, and patch version numbers.

Version(major, minor)

Creates a new instance of the `Version` class as a two-part package version using the specified major and minor version numbers.

Signature

```
public Version(Integer major, Integer minor)
```

Parameters

major

Type: Integer

The major version number.

minor

Type: Integer

The minor version number.

Version(major, minor, patch)

Creates a new instance of the `Version` class as a three-part package version using the specified major, minor, and patch version numbers.

Signature

```
public Version(Integer major, Integer minor, Integer patch)
```

Parameters

major

Type: Integer

The major version number.

minor

Type: Integer

The minor version number.

patch

Type: Integer

The patch version number.

Version Methods

The following are methods for `Version`. All are instance methods.

- **compareTo(version)**
  Compares the current version with the specified version.
- **major()**
  Returns the major package version of the of the calling code.

compareTo(version)

Compares the current version with the specified version.

Signature

```
public Integer compareTo(System.Version version)
```

Parameters

version

Type: System.Version

Return Value

Type: Integer

Returns one of the following values:

- zero if the current package version is equal to the specified package version
- an Integer value greater than zero if the current package version is greater than the specified package version
- an Integer value less than zero if the current package version is less than the specified package version

Usage

If a two-part version is being compared to a three-part version, the patch number is ignored and the comparison is based only on the major and minor numbers.

major()

Returns the major package version of the of the calling code.

Signature

```
public Integer major()
```

Return Value

Type: Integer

minor()

Returns the minor package version of the calling code.

Signature

```
public Integer minor()
```

Return Value

Type: Integer

patch()

Returns the patch package version of the calling code or `null` if there is no patch version.

Signature

```
public Integer patch()
```

DID THIS ARTICLE SOLVE YOUR ISSUE?
Let us know so we can improve!

Share your feedback

DEVELOPER CENTERS
- Heroku
- MuleSoft
- Tableau
- Commerce Cloud
- Lightning Design System
- Einstein
- Quip

POPULAR RESOURCES
- Documentation
- Component Library
- APIs
- Trailhead
- Sample Apps
- Podcasts
- AppExchange

COMMUNITY
- Trailblazer Community
- Events and Calendar
- Partner Community
- Blog
- Salesforce Admins
- Salesforce Architects