



Hibernate Validator 9.2.0.Beta1 - Jakarta Validation Reference Implementation

Migration Guide from 9.0

2026-10-13

Table of Contents

Introduction	1
Requirements	2
Artifacts	3
Configuration	4
API	5
SPI	6
Property node name providers	6
Constraint validator initialization	6
Behavior	7
CPF validation	7
Message parameter escaping enabled by default	7
Startup logging	7

Introduction

The aim of this guide is to assist you migrating an existing application using any version 9.0.x of Hibernate Validator to the latest of the 9.2.x series.



If you think something is missing or something does not work, please [contact us](#).

If you're looking to migrate from an earlier version, you should migrate step-by-step, from one minor version to the next, following the migration guide of [each version](#).

Requirements

The requirements of Hibernate Validator 9.2 are the same as those of Hibernate Validator 9.0.

Artifacts

The coordinates of Maven artifacts in Hibernate Validator 9.2 are the same as in Hibernate Validator 9.0.

The core engine now depends on `org.hibernate.accessor:hibernate-accessor`. Dependency managers resolve this transitively; applications managing their runtime class path manually must include it. The ASM and ByteBuddy accessor implementations are optional.

Configuration

The configuration properties in Hibernate Validator 9.2, in general, are backward-compatible with Hibernate Validator 9.0.

New configuration properties include `hibernate.validator.accessor_factory`, `hibernate.validator.message_interpolator` and `hibernate.validator.bean_meta_data_class_normalizer`. Existing programmatic setters take precedence over the corresponding properties. For components with built-in defaults, registering an implementation through a `BeanConfigurer` alone does not replace the default: use the component's setter or configuration property to select it.

API

The [API](#) in Hibernate Validator 9.2 is, in general, backward-compatible with Hibernate Validator 9.0.

SPI

The [SPI](#) in Hibernate Validator 9.2 is, in general, backward-compatible with Hibernate Validator 9.0.

Property node name providers

`PropertyNodeNameProvider#getName(Property)` is deprecated for removal. Implement `getName(Property, PropertyNodeNameProviderContext)` instead. The method's default implementation delegates to the old method, so existing implementations overriding `getName(Property)` continue to work.

Both methods are now default methods, so `PropertyNodeNameProvider` is no longer a functional interface. Replace lambda expressions and method references implementing this SPI with a class or anonymous class overriding the new method.

Constraint validator initialization

`HibernateConstraintValidatorInitializationContext` now declares `getBeanResolver()`.

Behavior

The behavior of Hibernate Validator 9.2 is, in general, backward-compatible with Hibernate Validator 9.0.

CPF validation

@CPF now accepts only the fully formatted XXX.XXX.XXX-XX representation or an unformatted sequence of 11 digits, with valid check digits in either case. Partially formatted values, including values with only a dash before the check digits, are rejected. Normalize such input to one of the supported representations before validation.

Message parameter escaping enabled by default

Message parameters added via `HibernateConstraintValidatorContext#addMessageParameter()` are now escaped by default. Characters with special meaning in message interpolation (`\`, `{`, `}`, `$`) are automatically escaped to prevent injection into message interpolation expressions.

If your custom constraint validators rely on message parameters containing unescaped interpolation expressions (e.g. nested `{paramName}` references or `${expression}` EL expressions), you can either:

- Use the new overload `addMessageParameter(String name, Object value, MessageParameterEscaping.NONE)` to disable escaping per call. This is a recommended alternative, **but** only for the parameter values that you trust.
- Set the `hibernate.validator.message_parameter_escaping` property to `none` to restore the previous behavior globally. Only consider this option if none of the validators add user input as parameters.

Startup logging

The Hibernate Validator version banner (HV000001) is now logged at `DEBUG` instead of `INFO`. Enable debug logging for `org.hibernate.validator.internal.util.Version` if you need this banner.