Meldeverfahren BUB – Unbedenklichkeitsbescheinigung

– Einheitliche Kernprüfung –

Betriebshandbuch

Autor: Verband der Ersatzkassen e.V. (vdek)

Datum: 25.09.2026

Dokumentenversion: 1.2.1

INHALT

1. Änderungsübersicht 3

2. Allgemeines 4

2.1 Voraussetzungen: 4

2.2 Lieferumfang: 4

2.3 Kernprüfungskonvention: 6

3 Einheitliche Schnittstellen 6

4 Einbindung der Kernprüfung 7

1. Änderungsübersicht

Version	Datum	Kap./Seite	Änderung
1.0.0	25.09.2023		Neuerstellung für BUB
1.1.0	31.05.2024	Kap. 4	Erweiterung in der Schnittstelle
1.2.0	11.09.2025	2.1, 2.2	Umstellung auf Java 11 und JAXB 4
1.2.1	25.09.2026	2.2	Aktualisierung der Basis- und JAXB-Bibliotheken

2. Allgemeines

Dieses Dokument ist die technische Anleitung für die Installation und den Betrieb der Komponente für die Kernprüfung im Verfahren BUB (Unbedenklichkeitsbescheinigung).

2.1 Voraussetzungen:

Als Laufzeit- / Entwicklungsumgebung wird Java Runtime Environment (JRE) oder Java Developer Kit (JDK) 11 vorausgesetzt. Weitere Komponenten oder Programme werden nicht benötigt.

2.2 Lieferumfang:

Die Kernprüfung wird in zwei unterschiedlichen Varianten bereitgestellt:

1. all_in_one Variante
2. single_jars Variante

In der all_in_one Variante werden alle benötigten Bibliotheken mit in die KernprüfungBUB.jar zusammengefasst, so dass man nur noch diese eine Java-Bibliothek laden muss, um die Kernprüfung vollständig verwenden zu können.

Die Version mit den single_jars hingegen beinhaltet alle benötigten Bibliotheken als separate einzelne Jar-Dateien. Somit muss man alle Bibliotheken einzeln laden, um die Kernprüfung verwenden zu können. Im Folgenden werden beispielhaft der Inhalte dieser Varianten zum besseren Verständnis aufgezeigt:

all_in_one

KernpruefungBUB-02.00.01.jar
KernpruefungBUB-02.00.01-javadoc.jar
KernpruefungBUB-02.00.01-sources.jar

single_jars

istack-commons-runtime.4.1.2.jar
istack-commons-runtime.4.1.2-javadoc.jar
istack-commons-runtime.4.1.2-sources.jar
jakarta.activation-api-2.1.4.jar
jakarta.activation-api-2.1.4 -javadoc.jar
jakarta.activation-api-2.1.4 -sources.jar
jakarta.xml.bind-api-4.0.5.jar
jakarta.xml.bind-api-4.0.5-javadoc.jar
jakarta.xml.bind-api-4.0.5-sources.jar
jaxb-core-4.0.9.jar
jaxb-core-4.0.9-javadoc.jar
jaxb-core-4.0.9-sources.jar
jaxb-runtime-4.0.9.jar
jaxb-runtime-4.0.9-javadoc.jar
jaxb-runtime-4.0.9-sources.jar
kernpruefung-api-01.00.00.jar
kernpruefung-api-01.00.00-javadoc.jar
kernpruefung-api-01.00.00-sources.jar
kernpruefung-xmlbasis-02.02.01.004.jar
kernpruefung-xmlbasis-02.02.01.004-sources.jar
KernpruefungBUB-02.00.01.jar
KernpruefungBUB-02.00.01-javadoc.jar
KernpruefungBUB-02.00.01-sources.jar

Folgende Komponenten werden in komprimierter Archivform (BUB.n.n.n.zip) ausgeliefert:

- Benutzer-Betriebshandbuch : Dieses Dokument
- Versionshistorie : Änderungsdokumentation
- Java-Doc : Entwicklerdokumentation Prüfmodul BUB
- Kernprüfprogramm : Alle java- und class-Dateien des Prüfmoduls
- Datensatzbeschreibung : aktueller Fehlerkatalog als Grundlage
- Freigabeerklärung : Erklärung über die erfolgte Qualitätssicherung
- XML-Schemata : aktuelle Schemata

2.3 Kernprüfungskonvention:

Alle benötigten Java-Sourcen, sowie deren Kompilate sind in folgenden Archiven enthalten:

- KernpruefungBUB-n.n.n.jar
- KernpruefungBUB-n.n.n-source.zip
- KernpruefungBUB-n.n.n-Javadoc.zip

Die Namenskonvention des Java-Archives setzt sich zusammen aus dem Namen **Kernpruefung**, dem Verfahrensnamen (in diesem Fall **BUB**) und schließlich aus der Versionsnummer **n.n.n**. Die ersten beiden Ziffern dieser Nummer entsprechen der Versionsnummer aus der Datensatzbeschreibung. Die letzte Ziffer ist für die Bereinigung von Fehlern vorgesehen. (Ziffer 0 = Urversion dieses Kernprüfprogramms für eine Version einer Datensatzbeschreibung).

3 Einheitliche Schnittstellen

Für alle einheitlichen Kernprüfungen wurden zwei Java-Interfaces entworfen, die von den einzelnen Kernprüfverfahren implementiert werden. Die beiden Schnittstellen sind im Java-Package **`kernpruefung`** zusammengefasst. Sie enthalten die Methoden für die Kernprüfung. Die Implementierung der Methoden wird in den unterschiedlichen Kernprüfverfahren realisiert.

Das Interface **`kernpruefung.Kernpruefung`** enthält die Methode **`pruefe(..)`**. Als Übergabeparameter wird der vollständige XML Datensatz(inklusiv Namespaces) und der vollständige Vorlaufsatz (VOSZ) übergeben (für die Prüfung im XML Verfahren kann der Vorlaufsatz entfallen). Es werden die Prüfungen aus der Datensatzbeschreibung bzw. des Fehlerkataloges durchgeführt. Das Ergebnis der Prüfung ist ein Rueckgabe-Objekt.

Im Interface **`kernpruefung.Rueckgabe`** sind Methoden für die Abfrage des Ergebnisses enthalten. Die Methode **`getReturnCode()`** gibt den Returncode als int-Wert zurück.

Es wurden folgende Returncodes festgelegt:

Returncode	Bedeutung

0	Kernprüfung fehlerfrei
1	Kernprüfung enthält Hinweise
2	Kernprüfung enthält Fehler
4	Kernprüfung beendet, ohne Prüfung (Abbruch)

Mit der Methode **`getRueckgabeMeldung()`** werden die Fehlerbausteine als String-Array zurückgegeben. Die Anzahl der Fehler oder Hinweise ist auf neun beschränkt.

Die String-Darstellung des Rueckgabe-Objektes erhält man mit der Methode **`toString()`**. Der Aufbau des Strings wird in der folgenden Tabelle beschrieben:

Stelle		Inhalt
von	bis	
1	1	Return-Code
2	2	Anzahl der Fehlerbausteine (max. 9)
3	78	Fehlerbaustein 1 (falls vorhanden)
79	154	Fehlerbaustein 2 (falls vorhanden)
155	230	Fehlerbaustein 3 (falls vorhanden)
231	306	Fehlerbaustein 4 (falls vorhanden)
307	382	Fehlerbaustein 5 (falls vorhanden)
383	458	Fehlerbaustein 6 (falls vorhanden)
459	534	Fehlerbaustein 7 (falls vorhanden)
535	610	Fehlerbaustein 8 (falls vorhanden)
611	686	Fehlerbaustein 9 (falls vorhanden)

4 Einbindung der Kernprüfung

Um die Kernprüfung in ein Verfahren einzubinden, ist es nötig die konkrete Schnittstellenimplementierung zu instanziieren.

Erzeugung einer Kernprüfer-Instanz zur aktuellen Version der Datensatzbeschreibung:

```
import com.vdek.agv.bub.kp.KernpruefungBUBImpl;
Kernpruefung kp = new KernpruefungBUBImpl();
```

Erzeugung einer Kernprüfer-Instanz zu einer bestimmten Version der Datensatzbeschreibung:

```
import com.vdek.agv.bub.kp.KernpruefungBUBImpl;
// version Versionsnummer, gegen die geprüft werden soll
Kernpruefung kp = new KernpruefungBUBImpl(version);
```

Alle Versionen, die von der Kernprüfer unterstützt, können über die statische Methode `getVersionen()` abgefragt werden.

Alternativ kann auch der Konstruktor mit mehreren Parametern verwendet werden:

```
import com.vdek.agv.bub.kp.KernpruefungBUBImpl;
//Parameters:
//        version Versionsnummer, gegen die geprüft werden soll
//        activeLogging Schalter der das Logging der Kernprüfung aktiviert(true) oder deaktiviert(false)
//        rcXML der Return-Code der bei Fehlern bei der Schemavalidierung verwendet werden soll
Kernpruefung kp = new KernpruefungBUBImpl(„01.01“,true,4);
```

Über die erzeugte Instanz kp können nun die Prüfungen mittels der Methode pruefe(..) durchgeführt werden.

Wichtig: In dem übergebenen Datensatz müssen sämtliche verwendete Namensraumpräfixe deklariert sein!

```
// im XML Verfahren kann vorlaufsatz null oder leer sein
Rueckgabe rueckgabe = kp.pruefe( datensatz, null);
```

Die Ergebnisse werden an das Rückgabeobjekt gereicht und können nun ausgewertet werden.

Zusammengefasst an einem Beispiel könnte die Einbindung folgendermaßen durchgeführt werden:

```
import kernpruefung.Kernpruefung;
import kernpruefung.Rueckgabe;
import com.vdek.agv.bub.kp.KernpruefungBUBImpl;

public class Beispiel
```

```
{
      public void tueWas()
      {
            String datensatz=
             "<?xml Version ="1.0" encoding ="ISO-8859-1" standalone…"
            + "<p:UB_AG Versionsnummer=… xmsns:…";

            Kernpruefung kp = new KernpruefungBUBImpl();

            Rueckgabe rueckgabe = kp.pruefe( datensatz, null);

            System.out.println( rueckgabe.getReturnCode() );
            System.out.println( rueckgabe.getRueckgabeMeldungen() );
            System.out.println( rueckgabe.toString() );
      }
}
```