

Medicare Code Editor Software

## **Java API guide**

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

If this product includes UB-04 information: Copyright 2021, American Hospital Association ("AHA"), Chicago, Illinois. Reproduced with permission. No portion of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without prior express, written consent of AHA.

# Table of Contents

• Medicare Code Editor Software Java API Guide .....	5
○ System requirements .....	5
○ Dependencies .....	5
○ Calling the MCE Java jar .....	6
○ Sample code .....	6
○ Class and enum information .....	8
▪ Class Mcerecord .....	8
• <i>Constructor</i> .....	8
• <i>Functions</i> .....	9
▪ Class MceDiagnosisCode .....	10
• <i>Constructor</i> .....	10
• <i>Functions</i> .....	10
▪ Class MceProcedureCode .....	11
• <i>Constructor</i> .....	11
• <i>Functions</i> .....	11
▪ Class MceOutput .....	12
• <i>Functions</i> .....	12
▪ Enum AgeConflictType .....	12
▪ Enum EditType .....	12
▪ Enum Edit .....	13
○ Additional information .....	15
▪ Diagnosis edit flags .....	15
▪ Procedure edit flags .....	16



# Medicare Code Editor Software Java API Guide

This document describes how to call the Medicare Code Editor (MCE) Software from a Java® application. It also notes the necessary dependencies required to execute the MCE Software. Suggested system requirements are also provided. The use of MCE software outside of a Java calling program is also possible but outside the scope of this document. There is also an included sources jar for MCE.

## System requirements

Oracle®, OpenJDK, or other Java® Version 8 or greater.

## Dependencies

**Table 1. Third Party Software**

Library	Version	Purpose	Information
protobuf-java	4.0.0-rc-2	Embedded content storage/processing speed optimization	Copyright (c) 2021 Google, Inc. All rights reserved License type: New BSD license URL: <a href="https://ptolemy.berkeley.edu/ptolemyII/ptII11.0/ptII/lib/protobuf-license.htm">https://ptolemy.berkeley.edu/ptolemyII/ptII11.0/ptII/lib/protobuf-license.htm</a>
gfc-base-api	3.4.9	Grouper Foundation Class Common grouper calling framework and common business objects to standardize grouper development	3M created open source binary URL: <a href="https://github.com/3mcloud/GFC-Grouper-Foundation-Classes">https://github.com/3mcloud/GFC-Grouper-Foundation-Classes</a>
slf4j-api	2.7	Logging	Copyright (c) 2004-2017 QOS.ch License Type: SLF4j (MIT License) URL: <a href="http://www.slf4j.org/license.html">http://www.slf4j.org/license.html</a>

## Calling the MCE Java jar

The MCE Java jar can be called using the following steps:

1. Create and populate the MceRecord.
2. Create MceComponent for processing.
3. Use the process() method on MceComponent and pass in MceRecord as a parameter for editing.
4. Once processed, use the getOutput() method on MceRecord to get the results of editing

## Sample code

This sample code illustrates the prior steps using hardcoded claim information.

```
import gov.cms.editor.mce.component.edit.Const;
import gov.cms.editor.mce.model.MceDiagnosisCode;
import gov.cms.editor.mce.model.MceProcedureCode;
import gov.cms.editor.mce.model.enums.Edit;
import gov.cms.editor.mce.model.enums.EditType;
import gov.cms.editor.mce.transfer.MceOutput;
import gov.cms.editor.mce.transfer.MceRecord;
import java.util.List;
import java.util.Map;

class MceComponentTest {
    public static void main(String args[]) {
        String dischargeDate = "20201130"; // Format yyyyMMdd
        Integer icdVersion = Const.ICD_10;
        MceDiagnosisCode admitDiagnosis = new MceDiagnosisCode("A001");
        Integer age = 25;
        Integer sex = Const.MALE;
        Integer dischargeStatus = 20;
        Integer lengthOfStay = 14;

        MceRecordBuilder mceRecordBuilder = MceRecord.builder();
        mceRecordBuilder.withDischargeDate(dischargeDate);
        mceRecordBuilder.withIcdVersion(icdVersion);
        mceRecordBuilder.withAdmitDiagnosis(admitDiagnosis);
        mceRecordBuilder.withAgeYears(age);
        mceRecordBuilder.withSex(sex);
        mceRecordBuilder.withDischargeStatus(dischargeStatus);
        mceRecordBuilder.withLengthOfStay(lengthOfStay);

        MceRecord mceRecord = mceRecordBuilder.build();
```

```
mceRecord.addCode(new MceDiagnosisCode("A001"));
mceRecord.addCode(new MceDiagnosisCode("U071"));
mceRecord.addCode(new MceDiagnosisCode("A1817"));
mceRecord.addCode(new MceDiagnosisCode("A34"));

mceRecord.addCode(new MceProcedureCode("0016070"));
// To invoke the component for processing with a MceRecord object
MceComponent mceComponent = new MceComponent();
mceComponent.process(mceRecord);

// Processing results will be stored within MceOutput // Obtain outputs
from MceRecord MceOutput outputs = mceRecord.getMceOutput();
Integer versionUsed = outputs.getVersionUsed();
EditType type = outputs.getEditType();
Map<Edit, Integer> editCounter = outputs.getEditCounter();
// Obtain edits from each code
List<MceDiagnosisCode> diagnosisCodes = mceRecord.getDiagnoses();
for (MceDiagnosisCode dxCode : diagnosisCodes) {
    List<Edit> editsOnCode = dxCode.getEdits(); // Alternatively use
dxCode.getEditsString() to return a String of flags
    // Do something
    System.out.println("Code: " + dxCode +
        "\nEdits: " + editsOnCode.toString());
}

List<MceProcedureCode> procedureCodes = mceRecord.getProcedures();
for (MceProcedureCode sgCode : procedureCodes) {
    List<Edit> editsOnCode = sgCode.getEdits(); // Alternatively use
sgCode.getEditsString() to return a String of flags
    // Do something
    System.out.println("Code: " + sgCode +
        "\nEdits: " + editsOnCode.toString());
}

// Example outputs printed to console
System.out.println("Version Used: " + versionUsed);
System.out.println("Edit Type: " + type.name()); // You can use
type.ordinal() to get its numbered value
System.out.println("Sex Conflicts: " +
editCounter.get(Edit.SEX_CONFLICT)); // Print the number of Sex Conflicts
encountered
}
}
```

## Class and enum information

### Class Mcerecord

#### Constructor

```
public MceRecord(String dischargeDate, Integer icdVersion, MceDiagnosisCode admitDiagnosis,
    Integer ageYears, Integer sex, Integer dischargeStatus, Integer lengthOfStay)
```

**Table 2. MCERecord constructor parameters**

Field	Type	Comments
dischargeDate	String	The discharge date on the claim in yyyyMMdd format
icdVersion	Integer	0 - ICD 10, 9 - ICD 9. Null if is blank or is missing.
admitDiagnosis	MceDiagnosisCode	Admit diagnosis code
ageYears	Integer	Valid range of values are between 0 - 140. Null if blank or is missing.
sex	Integer	1 - MALE, 2 - FEMALE. Null if is blank or is missing.
dischargeStatus	Integer	UB04 discharge status value. Null if is blank or is missing.
lengthOfStay	Integer	Length of stay of patient in days. Null if is blank or is missing.
diagnoses	List<MceDiagnosisCode>	A list of diagnosis codes as MceDiagnosisCode class. Use addCode function with MceDiagnosisCode as parameter. Principal diagnosis code should be added first.



Field	Type	Comments
procedures	List<MceProcedureCode>	A list of procedure codes as MceProcedureCode class. Use addCode function with MceProcedureCode as parameter.

## Functions

**Table 3. MCE record getters**

Function	Return Type	Comments
getDischargeDate	String	Return the discharge date that was inputted
getIcdVersion	Integer	Return the ICD version that was inputted
getAdmitDiagnosis	MceDiagnosisCode	Return the admit diagnosis that was inputted with editing information
getDiagnoses	List<MceDiagnosisCode>	Return a list of diagnoses that was added with editing information. Does not include admit diagnosis.
getPrincipalDiagnosis	MceDiagnosisCode	First code from the diagnosis list with editing information.
getSecondaryDiagnoses	List<MceDiagnosisCode>	Return a list of secondary diagnoses with editing information.
getProcedures	List<MceProcedureCode>	Return a list of procedures that was added with editing information.
getAgeYears	Integer	Return the age that was inputted.
getSex	Integer	Return the sex that was inputted.

Function	Return Type	Comments
getDischargeStatus	Integer	Return the discharge status that was inputted.
getLengthOfStay	Integer	Return the length of stay that was inputted.
getMceOutput	MceOutput	Obtain the outputs post processing
addCode	void	Call to add a Diagnosis or Procedure code to the record.

## Class MceDiagnosisCode

### Constructor

```
public MceDiagnosisCode(String value)
```

**Table 4. MceDiagnosisCode constructor parameter**

Field	Type	Comments
value	String	Diagnosis code value

### Functions

**Table 5. MceDiagnosisCode getters**

Function	Return Type	Comments
getValue	String	Return the code value
getEdits	List<Edit>	Return a list of edits that apply to the code if any.
getEditsString	String	Return the edits as a string of 0s and 1s based on Diagnosis Edit Flags (page <a href="#">15</a> ). 1 if code has edit, 0 if it doesn't.

Function	Return Type	Comments
getAgeConflictType	AgeConflictType	The type of age conflict encountered, if any.

## Class MceProcedureCode

### Constructor

```
public MceProcedureCode(String value)
```

**Table 6.** MceProcedureCode constructor parameter

Field	Type	Comments
value	String	Procedure code value

### Functions

**Table 7.** MceProcedureCode getters

Function	Return Type	Comments
getValue	String	Return the code value
getEdits	List<Edit>	Return a list of edits that apply to the code if any.
getEditsString	String	Return the edits as a string of 0s and 1s based on Procedure Edit Flags (page <a href="#">16</a> ). 1 if code has edit, 0 if it doesn't.

## Class MceOutput

### Functions

**Table 8. MceOutput getters**

Function	Return Type	Comments
getVersionUsed	Integer	The current version of the component
getEditType	EditType	The edit type encountered
getEditCounter	Map<Edit, Integer>	The count of the edits encountered

## Enum AgeConflictType

**Table 9. AgeConflictType enum**

Enum
NEWBORN (1), PEDIATRIC (2), MATERNITY (3), ADULT (4)

## Enum EditType

**Table 10. EditType enum**

Enum
NONE, PREPAYMENT, POSTPAYMENT, BOTH, INVALID_DISCHARGE_DATE

## Enum Edit

**Table 11. Edit enum**

Enum	Code Type	Comments
INVALID_CODE	BOTH	Count of any invalid diagnosis or procedure. Does not count admit diagnosis.
SEX_CONFLICT	BOTH	Count of any diagnosis or procedure with sex conflict.
AGE_CONFLICT	DIAGNOSIS	
QUESTIONABLE_ADMISSION	DIAGNOSIS	
MANIFESTATION_AS_PDX	DIAGNOSIS	
NONSPECIFIC_PDX	DIAGNOSIS	
E_CODE_AS_PDX	DIAGNOSIS	
UNACCEPTABLE_PDX	DIAGNOSIS	
DUPLICATE_OF_PDX	DIAGNOSIS	Count of secondary diagnoses that match with the principal diagnosis. Does not include admit diagnosis.
MEDICARE_IS_SECONDARY_PAYER	DIAGNOSIS	
REQUIRES_SDx	DIAGNOSIS	
NONSPECIFIC_OR	PROCEDURE	
OPEN_BIOPSY	PROCEDURE	
NON_COVERED	PROCEDURE	
BILATERAL	PROCEDURE	
LIMITED_COVERAGE_LVRS	PROCEDURE	
LIMITED_COVERAGE	PROCEDURE	Used in I10 to capture any and all limited coverage related edits
LIMITED_COVERAGE_LUNG_TRANSPLANT	PROCEDURE	
QUESTIONABLE_OBSTETRIC_ADMISSION	PROCEDURE	
LIMITED_COVERAGE_COMBINATION_HEART_LUNG_TRANSPLANT	PROCEDURE	

Enum	Code Type	Comments
LIMITED_COVERAGE_HEART_TRANSPLANT	PROCEDURE	
LIMITED_COVERAGE_HEART_IMPLANT	PROCEDURE	
LIMITED_COVERAGE_INTESTINE_MULTI_VISCERAL_TRANSPLANT	PROCEDURE	
LIMITED_COVERAGE_LIVER_TRANPSLANT	PROCEDURE	
LIMITED_COVERAGE_KIDNEY_TRANSPLANT	PROCEDURE	
LIMITED_COVERAGE_PANCREAS_TRANSPLANT	PROCEDURE	
LIMITED_COVERAGE_ARTIFICIAL_HEART	PROCEDURE	
INVALID_ADMIT_DX	NONE	Code type is intentionally set to NONE for internal use.
INVALID_AGE	NONE	
INVALID_SEX	NONE	
INVALID_DISCHARGE_STATUS	NONE	
TYPE_OF_AGE_CONFLICT	DIAGNOSIS	
INVALID_POA	DIAGNOSIS	For future use.
WRONG_PROCEDURE_PERFORMED	DIAGNOSIS	
INCONSISTENT_WITH_LENGTH_OF_STAY	PROCEDURE	
UNSPECIFIED	DIAGNOSIS	
DX_UNUSED_1	DIAGNOSIS	
DX_UNUSED_2	DIAGNOSIS	
DX_UNUSED_3	DIAGNOSIS	
DX_UNUSED_4	DIAGNOSIS	
DX_UNUSED_5	DIAGNOSIS	
SG_UNUSED_1	PROCEDURE	
SG_UNUSED_2	PROCEDURE	
SG_UNUSED_3	PROCEDURE	

See the [Definition of Medicare Edits for additional details on these edits.](#)

## Additional information

### Diagnosis edit flags

**Table 12.      Diagnosis edit flags**

Index	Edit Description
0	Invalid diagnosis code
1	Sex conflict
2	Age conflict
3	Questionable admission
4	Manifestation code as principal diagnosis
5	Nonspecific principal diagnosis
6	E-code as principal diagnosis
7	Unacceptable principal diagnosis
8	Duplicate of principal diagnosis
9	Medicare is secondary payer
10	Requires secondary diagnosis
11	Type of age conflict: 0 - No age conflict 1 - Newborn 2 - Pediatric 3 - Maternity 4 - Adult
12	POA indicator invalid or missing (for future use)
13	Wrong procedure performed
14	Unspecified
15	Unused
16	Unused
17	Unused
18	Unused

Index	Edit Description
19	Unused

## Procedure edit flags

**Table 13. Procedure edit flags**

Index	Edit Description
0	Invalid procedure code
1	Sex conflict
2	Nonspecific O.R. procedure
3	Open biopsy check
4	Non-covered procedure
5	Bilateral procedure
6	ICD9 - Limited coverage, lung volume reduction surgery ICD10 - Represent all limited coverage procedures
7	ICD9 - Limited coverage, lung transplant ICD10 - Questionable obstetric admission
8	Limited coverage, combination heart and lung transplant
9	Limited coverage, heart transplant
10	Limited coverage, implant of heart assist system
11	Limited coverage, intestine or multi-visceral transplant
12	Limited coverage, liver transplant
13	Limited coverage, kidney transplant
14	Limited coverage, pancreas transplant
15	Limited coverage, artificial heart
16	Procedure inconsistent with length of stay
17	Unused
18	Unused



<b>Index</b>	<b>Edit Description</b>
19	Unused