Zyklomatische Komplexität (CC)

CC = Nr. Regionen = 6CC = Nr. Kanten - Nr. Knoten + 2 = 14 - 10 + 2 = 6

findById serviceSave def findById(id):... def serviceSave(chord):... def save(selectedChordId):.. return id return chord if selectedChordId is None: selectedChordId is not None) selectedChordId is None chord = findById(selectedChordId).orElseThrow() if overrideMetadata: overrideMetadata Call.orElseThrow chord += 1 (not overrideMetadata) overrideMetadata = False if stemType is not None: stemType is not None $chord \mathrel{+}= 1$ (stemType is None) stemType = None if overrideStemMetadata overrideStemMetadata chord += 1 (not overrideStemMetadata) overrideStemMetadata = False if dotCount is not None: dotCount is not None $chord \mathrel{+}= 1$ (dotCount is None) dotCount = None serviceSave(chord) calls

cfg.py

serviceSave

CC = Nr. Bedingungen + 1 = 5 + 1 = 6

Indiviudelle Pfaden

TC01:
$$1(A) - 2(A) - 3(A) - 4(A) - 5(A)$$

TC02:
$$1(A) - 2(A) - 3(A) - 4(A) - 5(B)$$

TC03:
$$1(A) - 2(A) - 3(A) - 4(B) - 5(A)$$

• • •

TC29:
$$1(B) - 2(B) - 3(B) - 4(A) - 5(A)$$

TC30:
$$1(B) - 2(B) - 3(B) - 4(A) - 5(B)$$

TC31:
$$1(B) - 2(B) - 3(B) - 4(B) - 5(A)$$

TC32:
$$1(B) - 2(B) - 3(B) - 4(B) - 5(B)$$

Total: $2^5 = 32$.

TC24

Input: stemType = Half, dotCount = 2, stemMetadata = null, chordId = null, metadata = null Expected: stemType = Half, dotCount = 2, stemMetadata = null, chordId = null, metadata = null

TC25

Input: stemType = Half, dotCount = 0, stemMetadata = null, chordId = null, metadata = "RandomMetadata" Expected: stemType = Half, dotCount = 0, stemMetadata = null, chordId = null, metadata = "RandomMetadata"

TC28

Input: stemType = Half, dotCount = 0, stemMetadata = "RandomStemMetadata", chordId = null, metadata = null Expected: stemType = Half, dotCount = 0, stemMetadata = "RandomStemMetadata", chordId = null, metadata = null

Coverage: 3 / 32 = 9,3%