Bookkepper test

Luca Falasca

5 settembre 2023

Indice

Elenco delle figure

Elenco delle tabelle

1 Introduction

2 Bookkeeper

2.1 Journal

2.1.1 Descrizione

listJournalIds Lists all journal IDs filtered by a specified journal ID filter.
This method scans the given directory containing journal log files and extracts journal IDs based on the provided filter. If no filter is provided, all journal IDs present in the directory are returned.

Input:

- journalDir journal dir : The directory containing journal log files.
- filter journal id filter

Output: list of filtered ids

2.1.2 Category Partition

journalDir

- {Directory contenente file di log}
- {Directory contenente file di log e altri file}
- {Directory contenente file non di log}
- {path non esistente}
- {Path di un file}
- null

Siccome la variabile journalDir è il path di una directory, ho partizionato il dominio in base al contenuto della directory e al suo effettivo utilizzo, rendendo la scelta delle partizioni una conseguenza del fatto che il metodo lavora su file di log.

JournalIdFilter

- Filtro esistente
- Filtro sempre True
- Filtro sempre False
- Filtro inesistente
- null

2.1.3 Boundary Analysis

Andiamo a definire per ogni partizione i Boundary values

journalDir

- {Directory contenente 1 file di log}
- {Directory contenente 1 file di log e 1 file di testo}
- {Directory contenente 1 file di testo}
- {path non esistente}
- {Path di un file di log}
- null

JournalIdFilter

- JournalRollingFilter – Questo filtro è l'unico filtro esistente utilizzato nell'applicazione, tuttavia siccome viene utilizzato in un contesto molto specifico dell'applicazione e non è un filtro generico, l'ho rimpiazzato con una sua versione semplificata più generale. Altrimenti non sarebbe stato adatto ad un test di unità, ma sarebbe stato più un test di integrazione
- Filtro sempre True
- Filtro sempre False
- MyFilter – Questo filtro è un filtro personalizzato creato appositamente per questo test che va a filtrare i journal in base al loro nome, in particolare se il journalId è ¿ 0 viene accettato, altrimenti no. Questo filtro ha lo scopo di testare come si comporta il sistema in caso di definizione di un nuovo filtro non già esistente nel sistema e quindi non è importante il tipo di filtraggio che fa.
- null

Siccome i due parametri di input sono abbastanza scorrelati tra loro ha più senso adottare un approccio unidimensionale piuttosto che uno multidimensionale che sarebbe più adatto quando ci sono delle interazioni forti e che portano alla necessità di testare tutte le combinazioni tra i parametri. Inoltre avendo un approccio multidimensionale si finirebbe probabilemente per avere molti test non rilevanti e vanno a coprire scenari già coperti, e quindi sarebbero inutili.
Enumeriano ora i casi di test derivanti da quest prima analisi (Tabella 1)

2.1.4 Adequacy Control Flow

Ora per verificare l'adeguatezza dei casi di test, vado a definire dei critedi di adeguatezza.

Criteri di adeguatezza black box

documentazione a disposizione:
javadoc:

```
/**
* List all journal ids by a specified journal id filer.
*
* @param journalDir journal dir
* @param filter journal id filter
* @return list of filtered ids
*/
```

documentazione apache: Journals A journal file contains BookKeeper transaction logs. Before any update to a ledger takes place, the bookie ensures that a transaction describing the update is written to non-volatile storage. A new journal file is created once the bookie starts or the older journal file reaches the journal file size threshold.

Possiamo dedurre da queste documentazioni che questo metodo ricava gli id dei journal in una determinata cartella, filtrandoli in base ad un filtro specificato, se non viene specificato nessun filtro, allora vengono restituiti tutti gli id dei journal presenti nella cartella.

La Figura 1 mostra un diagrammma funzionale del metodo listJournalIds. A partire da questo definisco i seguenti criteri di adeguatezza:

- almeno un test con un filtro
- almeno un test senza filtro
- almeno un test senza journal nella cartella
- almeno un test con almeno un journal nella cartella

In questo caso i criteri di adeguatezza sono già stati soddisfatti dai casi di test precedentemente descritti.

Dal diagramma precedente (Figura 1) possiamo ricavare un control flow graph (Figura 5), per poi utilizzare dei criteri di copertura basati sul control flow

Dato che ho un approccio black box e quindi non sto utilizzando il codice sorgente per verificare l'adeguatezza dei casi di test, ipotizzo in base agli input se un caso di test andrà a coprire un certo arco del CFG o meno. Dato che in questa fase sto avendo un approccio black box, e quindi basato sulla funzionalità del metodo, eviterò di utilizzare la coverage come parametro di adeguatezza, in quanto non è possibile ricavare dall'output del metodo se un certo arco è stato coperto o meno, ma solo se il risultato è corretto oppure no. Quindi valuterò la copertura del CFG solo in una fase successiva, quando prenderò in considerazione anche il codice sorgente (approccio whitebox), utilizzando la coverage come parametro di adeguatezza. Quello che farò in questa fase è quindi valutare se i casi di test che ho definito in precedenza coprono tutti gli archi del CFG, in base alla mia conoscenza del metodo e dei suoi input, e nel caso aggiungere quelli mancanti.

Arco (1,2) $\rightarrow$ textgreater Tutti i test che hanno almeno un file di log coprono questo caso, quindi non è necessario aggiungere ulteriori test Arco (2,3) - Tutti i test che hanno almeno un file di log e un filtro non null coprono questo caso, quindi non è

necessario aggiungere ulteriori test Arco (2,4) - Questo arco non è coperto perchè manca un test che abbia almeno un file di log e il filtro null. Basta quindi aggiungere il test necessario (Figura 2)

Arco (3,4) - Tutti i test che hanno almeno un file di log e un filtro non null coprono questo caso, quindi non è necessario aggiungere ulteriori test

è stato necessario quindi aggiungere un caso di test e quindi la attuale test suite è descritta nella Tabella 5

Ora andiamo a valutare la coverage ottenuta con jacoco con i casi di test definiti fino ad ora (Figura **??**)

Come si può vedere dalla figura **??**, la statement coverage è del 100% (colonna missed instruction), e quindi tutte le linee di codice del metodo sono state eseguite

Invece la condition coverage (colonna missed branches) e del 91%, ed è quindi migliorabile. Andando a vedere in dettaglio il report, si può notare che la condizione che non è stata coperta è una delle 4 combinazioni dell'if a riga 106 come si vede dalla figura **??**

La condizione presa in esame è la seguente: logFiles == null —— logFiles.length == 0

il branch mancato è quello dove logFiles == null, questo perchè se il parametro di input journalDir è null non esegue proprio il metodo listFiles()

questo è dovuto al fatto che il path non era esistente e quindi il metodo tornava null

Andiamo quindi a definire un caso di test che copra questo branch Per farlo andiamo semplicemente basta aggiungere un category partition che copre il caso di una cartella esistente ma vuota (Tabella 4)

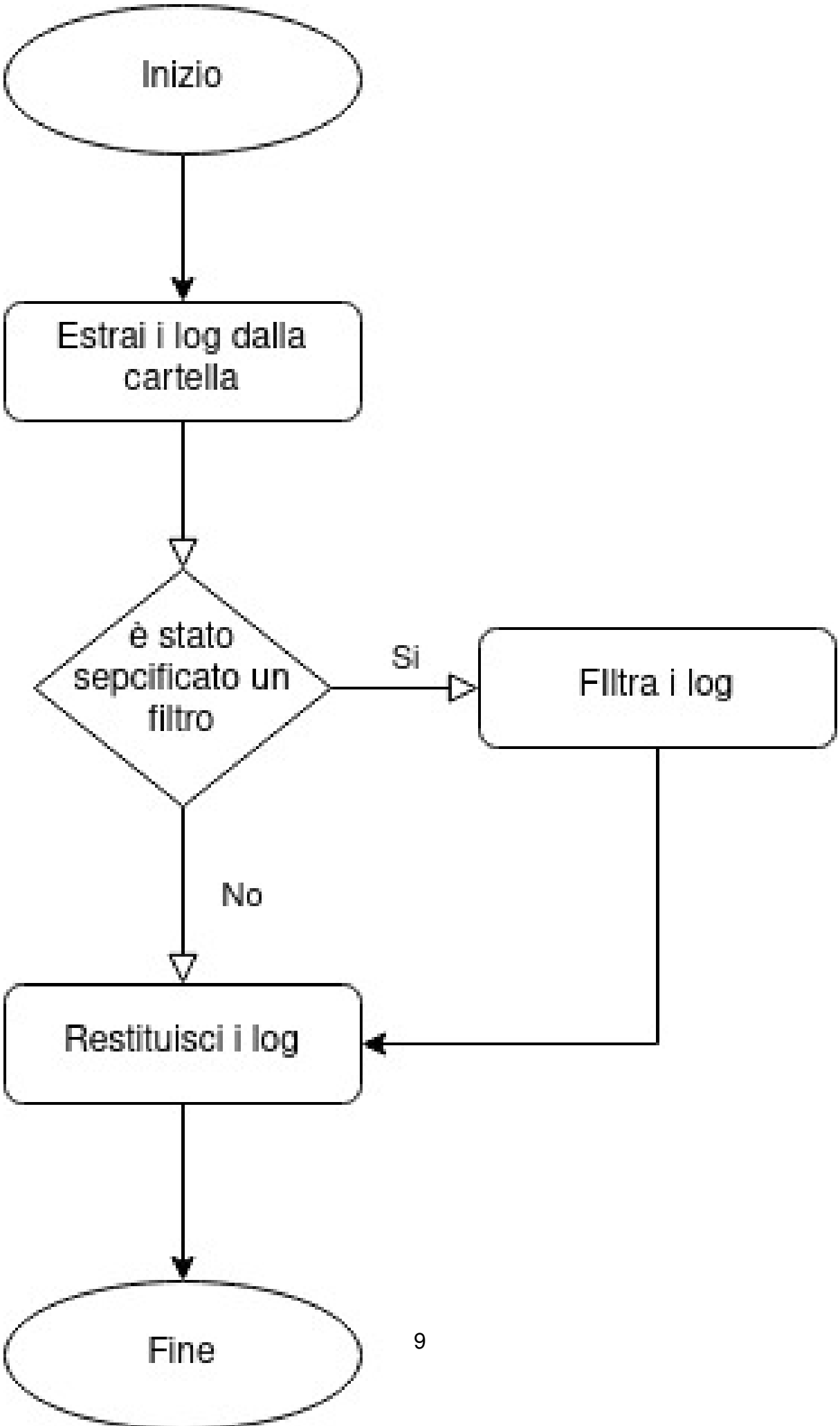
Ora quindi l'insieme dei test è il seguente

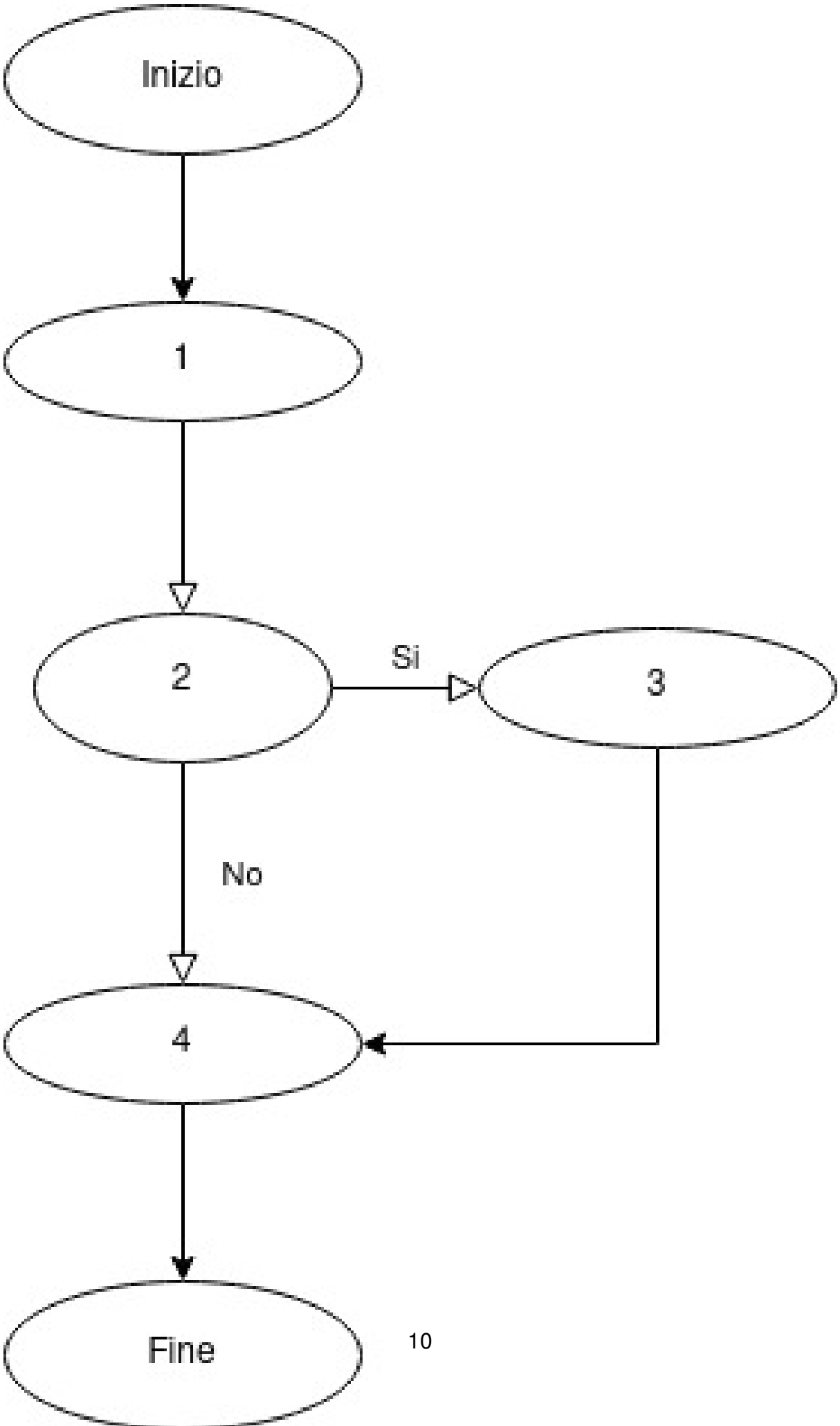
e grazie a questo test anche la condition coverage è del 100% (Figura **??**), con un aumento del 9

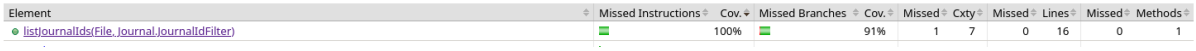
2.1.5 Adequacy Data Flow

Andiamo ora ad fare un lavoro di adeguatezza del dataflow utilizzando il framework badua per calcolare la all-uses coverage

journalDir	JournalIdFilter	Risultato Atteso
Directory contenente 1 file di log	JournalRollingFilter	...
Directory contenente 1 file di log	Filtro sempre True	...
Directory contenente 1 file di log	Filtro sempre False	...
Directory contenente 1 file di log	MyFilter	...
Directory contenente 1 file di log	null	Exception
Directory contenente 1 file di log e 1 file di testo	JournalRollingFilter	...
Directory contenente 1 file di log e 1 file di testo	Filtro sempre True	...
Directory contenente 1 file di log e 1 file di testo	Filtro sempre False	...
Directory contenente 1 file di log e 1 file di testo	MyFilter	...
Directory contenente 1 file di log e 1 file di testo	null	Exception
Directory contenente 1 file di testo	JournalRollingFilter	...
Directory contenente 1 file di testo	Filtro sempre True	...
Directory contenente 1 file di testo	Filtro sempre False	...
Directory contenente 1 file di testo	MyFilter	...
Directory contenente 1 file di testo	null	Exception
path non esistente	JournalRollingFilter	[ ]
path non esistente	Filtro sempre True	[ ]
path non esistente	Filtro sempre False	[ ]
path non esistente	MyFilter	[ ]
path non esistente	null	[ ]
Path di un file di log	JournalRollingFilter	...
Path di un file di log	Filtro sempre True	...
Path di un file di log	Filtro sempre False	...
Path di un file di log	MyFilter	...
Path di un file di log	null	Exception
null	JournalRollingFilter	Exception
null	Filtro sempre True	Exception
null	Filtro sempre False	Exception
null	MyFilter	Exception
null	null	Exception
Directory vuota	null	[ ]


Inizio
Estrai i log dalla cartella
è stato sepcificato un filtro
Si
FIltra i log
No
Restituisci i log
Fine


Inizio
1
2
Si
3
No
4
Fine



Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods
listJournalIds(File, Journal.JournalIdFilter)		100%		91%	1	7	0	16	0	1

Figura 3: Jacoco coverage dilistJournalIds

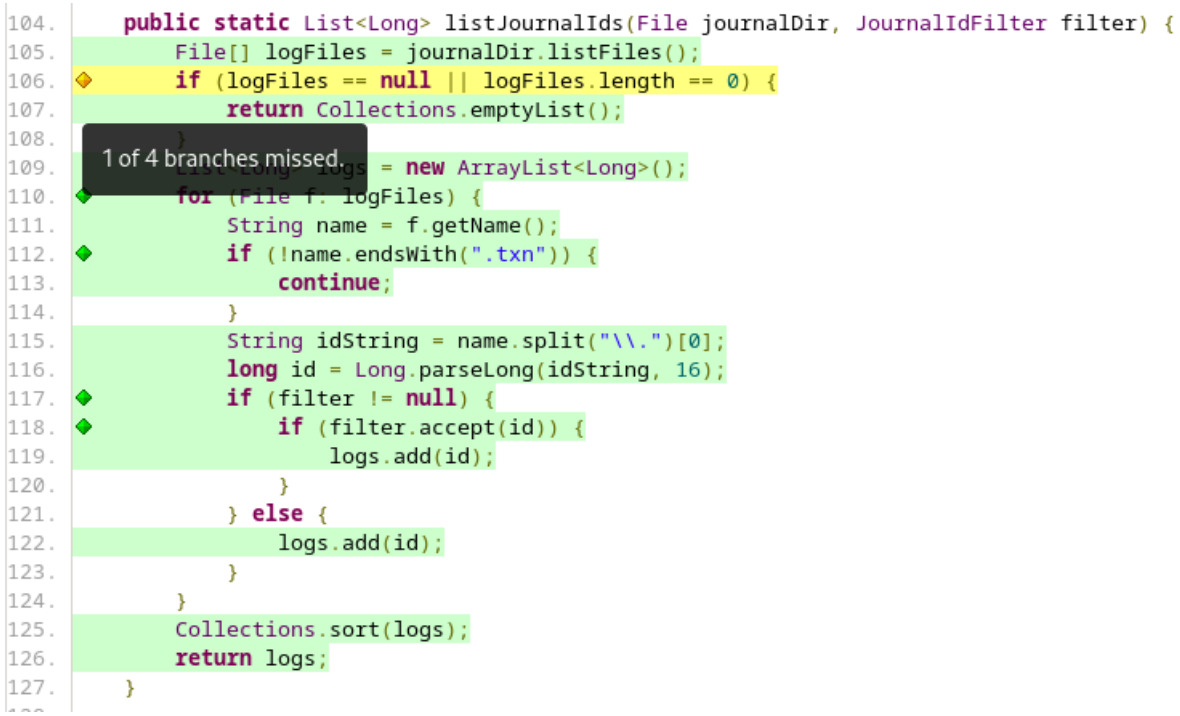
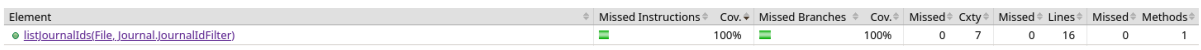


Figura 4: Jacoco coverage dilistJournalIds



Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods
listJournalIds(File, Journal.JournalIdFilter)		100%		100%	0	7	0	16	0	1

Figura 5: Jacoco coverage dilistJournalIds

Tabella 1: Test Suite derivante dalla category partition e dalla boundary analysis

journalDir	JournalIdFilter	Risultato Atteso
Directory contenente 1 file di log	JournalRollingFilter	[1]
Directory contenente 1 file di log e 1 file di testo	Filtro sempre True	[1]
Directory contenente 1 file di testo	Filtro sempre False	[ ]
path non esistente	MyFilter	[ ]
Path di un file di log	null	Exception
null	null	Exception

Tabella 2: Test per coprire l'arco (2,4)

journalDir	JournalIdFilter	Risultato Atteso
Directory contenente 1 file di log	null	Exception

Tabella 3: Casi di test aggiornati per coprire l'arco (2,4)

journalDir	JournalIdFilter	Risultato Atteso
Directory contenente 1 file di log	JournalRollingFilter	[1]
Directory contenente 1 file di log e 1 file di testo	Filtro sempre True	[1]
Directory contenente 1 file di testo	Filtro sempre False	[ ]
path non esistente	MyFilter	[ ]
Path di un file di log	null	Exception
null	null	Exception
Directory contenente 1 file di log	null	Exception

Tabella 4: Test per aumentare la condition coverage

journalDir	JournalIdFilter	Risultato Atteso
Directory vuota	null	[ ]

Tabella 5: Casi di test aggiornati per aumentare la condition coverage

journalDir	JournalIdFilter	Risultato Atteso
Directory contenente 1 file di log	JournalRollingFilter	[1]
Directory contenente 1 file di log e 1 file di testo	Filtro sempre True	[1]
Directory contenente 1 file di testo	Filtro sempre False	[ ]
path non esistente	MyFilter	[ ]
Path di un file di log	null	Exception
null	null	Exception
Directory contenente 1 file di log	null	Exception
Directory vuota	null	[ ]