Wiktoria Weichbrodt

Nr indeksu: 249609

Grupa: K00-28a (CZW/TP)

Sprawozdanie z interfejsów Closeable i Iterable

Interfejs to zestaw metod bez ich implementacji (bez kodu definiującego zachowanie metody). Właściwa implementacja metod danego interfejsu znajduje się w klasie implementującej dany interfejs. Interfejsy w bardzo prosty sposób ułatwiają różnego rodzaju integrację różnych fragmentów kodu.

Interfejs Closeable jest źródłem lub miejscem docelowym danych, które można zamknąć. Wywoływana jest metoda close w celu zwolnienia zasobów przechowywanych przez obiekt (takich jak otwarte pliki).

Closeable został wprowadzony w JDK5 i teraz wygląda następująco:

```
public interface Closeable extends AutoCloseable {

    public void close() throws IOException;

}
```

AutoCloseable został wprowadzony wraz z JDK7 i wygląda następująco:

```
public interface AutoCloseable {

    void close() throws Exception;

}
```

Closeable ma pewne ograniczenia, ponieważ może jedynie zgłaszać IOException, i nie można go zmienić bez zerwania odziedziczonego kodu(legacy code). Wprowadzono więc funkcję AutoCloseable, która może zgłaszać Exception.

Użyta metoda zamyka wywołujący obiekt, zwalniając wszelkie zasoby, które może on posiadać. AutoCloseable został specjalnie zaprojektowany do pracy z instrukcjami try-with-resources. Jest wywoływany automatycznie na końcu instrukcji try-with-resources, co eliminuje potrzebę jawnego wywoływania close(). Automatyczne zamykanie strumienia zapewnia jego prawidłowe zamknięcie, gdy nie jest już potrzebne, zapobiegając w ten sposób wyciekowi pamięci i innym problemom. Tylko obiekty klas, które implementują AutoCloseable mogą być zarządzane przez try-with-resources.

ABBYY PDF Transformer+ Click here to buy www.ABBYY.com

Natomiast implementacja interfejsu Iterable pozwala obiektowi stać się celem instrukcji „for-each loop”, inaczej zwaną pętlą iteracyjną, umożliwiającą przeglądnięcie wszystkich elementów należących do danej kolekcji. Jest to zdefiniowane w pakiecie java.lang i zostało wprowadzone w Javie 5.

```
public interface Iterable<T> {

 public Iterator<T> iterator();

}
```

Iterable<T>, gdzie parametr typu T reprezentuje typ elementów zwracanych przez iterator.

Iterable Interfejs Java ma tylko jedną metodę o nazwie iterator(). Ta metoda musi zwrócić Iterator Java, którego można użyć do iteracji (czynność powtarzania tej samej operacji w pętli z góry określoną liczbę razy lub aż do spełnienia określonego warunku) elementów obiektu implementującego interfejs Iterable.

Ilustrują to dwa poniższe fragmenty kodu, które mają tę samą funkcjonalność, używając odpowiednio pętli for i Iteratora:

```
for (int i = 0; i < stringList.size(); i++) {

   System.out.println(stringList [i]);

}

Iterator<String> iter = stringList.iterator();

while (iter.hasNext()) {

   System.out.println(iter.next());

}
```

Korzysta się z iteratora w momencie gdy nasza klasa przechowuje większą liczbę takich samych elementów, mówiąc inaczej jest kolekcją jakiś elementów. Implementacja interfejsu Iterable<T> znacznie upraszcza obsługę takiej klasy. W przypadku Closeable używa się go, aby zamknąć zasób i uwolnić wszelkie powiązane z nim zasoby podstawowe. Jeśli nie zamkniesz zasobów, których używasz, ostatecznie ich zabraknie. Zamykanie zasobów nie zawsze jest obsługiwane przez moduł wyrzucania elementów bezużytecznych, dlatego może prowadzić do marnowania pamięci.