Iterator Design Pattern

התבנית העיצובית של Iterator היא תבנית עיצוב מסוג התנהגות, היא מאפשרת לנו לסנן או לעבור על אוסף של אובייקטים מבלי לחשוף את המבנה הפנימי של האוסף (עקרון האינקפסולציה).

התבנית מאפשרת לנו מעבר על איברי אוסף מסויים מבלי לדעת איך האוסף בנוי מאחורי הקלעים. בעזרת זה אנחנו מקבלים קוד נקי יותר וקל לתחזוקה.

בדרך כלל נגדיר את הממשק `Iterator` שמגדיר את הפעולות הבסיסיות של מעבר על אוסף(כמו `next` , `hasNext`), ואת הממשק `Iterable` שמגדיר את הפעולות שמחזירות איטרטור.

> בשפת Java יש תמיכה מובנית בתבנית העיצוב של Iterator בעזרת הממשק `Iterable` והממשק `Iterator` .

דוגמה לשימוש ב- Iterator

נניח שיש לנו רשימת השמעה, ואחנו רוצים דרך קלה לעבור על הרשימה, נשתמש בתבנית העיצוב של Iterator כדי להסתיר את המימוש של הרשימה מאיתנו.

בדגומה הבאה יש לנו:

- `SongIterator` - ממשק שמגדיר את דרך המעבר שלנו על הרשימה (ממשק Iterator).
- `ArrayListSongIterator` - מחלקה שמממשת את האיטרטור עבור רשימת שירים מסוג ArrayList.
- `Playlist` - ממשק שמגדיר את הפעולות של רשימת ההשמעה (ממשק Iterable).
- `ArrayListPlaylist` - מחלקה שמממשת את רשימת ההשמעה מסוג ArrayList.

```java
// Iterator interface
interface SongIterator {
    boolean hasNext();
    Song next();
}

// Concrete iterator for an ArrayList-based playlist
class ArrayListSongIterator implements SongIterator {
    private ArrayList<Song> songs;
    private int index;

    public ArrayListSongIterator(ArrayList<Song> songs) {
        this.songs = songs;
        this.index = 0;
    }

    @Override
    public boolean hasNext() {
        return index < songs.size();
    }

    @Override
    public Song next() {
        return songs.get(index++);
    }
}

// Playlist interface(Iterable)
interface Playlist {
    SongIterator createIterator();
}

// Concrete playlist implementation using ArrayList
class ArrayListPlaylist implements Playlist {
    private ArrayList<Song> songs;

    public ArrayListPlaylist() {
        this.songs = new ArrayList<>();
```

```java
    }

    public void addSong(Song song) {
        songs.add(song);
    }

    @Override
    public SongIterator createIterator() {
        return new ArrayListSongIterator(songs);
    }
}

// Usage
Playlist playlist = new ArrayListPlaylist();
playlist.addSong(new Song("Song 1"));
playlist.addSong(new Song("Song 2"));

SongIterator iterator = playlist.createIterator();
while (iterator.hasNext()) {
    Song song = iterator.next();
    System.out.println("Playing: " + song.getTitle());
}
```