# Geliştirici Bilgileri

Türker Öztürk

https://github.com/turkerozturk

Proje Adı: `eventsporistanbulscraper`

# Sorumluluk

Geliştirici, yazılımın kullanılmasından kaynaklı oluşabilecek problemlerden ve veri eldesinin doğruluğundan sorumlu tutulamaz.

# Uygulama Lisansı

Atıf yapıldığı sürece, eserin değiştirilmeden ticari ya da gayri-ticari dağıtım hakkını verir.

https://creativecommons.org/licenses/by-nd/4.0/deed.tr

Atıf-Türetilemez 4.0 Uluslararası CC BY-ND 4.0

# Derleme

Proje klasöründe komut satırından aşağıdaki komut çalıştırılır. Sistemden maven kurulu olmasına gerek yoktur çünkü mvnw dosyası o işi halleder.

```
$ mvnw clean install
```

`target` klasörü oluşur.

Linux üzerinde çalıştırılacaksa `mvnw` dosyasına executable yetkisi verilmeli ve komutun başında `./` olmalıdır.

# Çalıştırma

Klasörün içindeki JAR dosyası dışında kalan dosyalar gereksizdir ve silinebilir. Aşağıdaki komut çalıştırıldığında uygulama çalışacak ve birkaç dakika içinde veritabanını internetteki verilerle doldurmuş olacak. İşlemler yapılırken konsola çıktı verdiğinden, hangi koşu verisini aldığını görmek mümkün. Veritabanı, JAR dosyasının çalıştırıldığı klasörde oluşur. İşlem sürecinde geçici dosyalar oluşturabilir, işlem bittiğinde geçici dosyalar kaybolur.

```
$ java -jar eventsporistanbulscraper-1.0-SNAPSHOT.jar
```

# Bağımlılıklar

## SQLite Kütüphanesi

*pom.xml*

```
<!-- https://mvnrepository.com/artifact/org.xerial/sqlite-jdbc -->
<dependency>
  <groupId>org.xerial</groupId>
  <artifactId>sqlite-jdbc</artifactId>
  <version>3.44.1.0</version>
</dependency>
```

## Jsoup Kütüphanesi

*pom.xml*

```
<!-- https://mvnrepository.com/artifact/org.jsoup/jsoup -->
<dependency>
  <groupId>org.jsoup</groupId>
  <artifactId>jsoup</artifactId>
  <version>1.17.2</version>
</dependency>
```

*EventScraperWithSelenium.java*

```
assert pageSource != null;
Document doc = Jsoup.parse(pageSource);
```

## Selenium Kütüphanesi

*pom.xml*

```
<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
<dependency>
  <groupId>org.seleniumhq.selenium</groupId>
  <artifactId>selenium-java</artifactId>
  <version>4.25.0</version>
</dependency>
```

### Driver Dosyaları

Selenium kullanmak için https://googlechromelabs.github.io/chrome-for-testing/ sayfasından uygun arşiv dosyası indirilir.

Örneğin: https://storage.googleapis.com/chrome-for-testing-public/129.0.6668.89/win32/chrome-win32.zip

### Websayfasının Yüklenmesi

Driver executable dosyasının yeri kodda belirtilir.

*EventScraperWithSelenium.java*

```
String pathToChromeDriver =
"C:\\PROGRAMMING\\SELENIUM\\CHROMEDRIVERS\\129\\chromedriver-win32\\chromedriver.exe";
System.setProperty("webdriver.chrome.driver", pathToChromeDriver);
WebDriver driver = new ChromeDriver();
driver.get("https://event.spor.istanbul/eventresults.aspx");
```

driver.get metodu ile scrape etmek istediğimiz websayfasına gitmiş oluyoruz. Browser penceresi açılacak ve sayfa yüklenmiş olarak karşımızda görünecek

Browserlar arasından Chrome seçtik, işletim sistemi Windows, 32 bit olan. İşletim sistemimiz Windows 11 64 bit olmasına rağmen böyle seçtik. Sürüm seçimi önemli, hata verirse hata mesajında bir sayısal sürüm yazar, bilgisayarda kurulu olan Chromw uygulaması ve Java'daki selenium kütüphanesi sürüm olarak birbiriyle uyumlu olmalıdır.

# Oluşacak Veritabanı

*EventScraper.java*

```
public static final String DB_URL = "jdbc:sqlite:runningResults.sqlite";
```

# Koşu Verilerini İndirme Planı

Sayfa incelendiğinde teknik olarak bütün koşu verisini indirme yöntemi şöyle belirlendi:

*EventScraperWithSelenium.java*

```
String ddlEvents = "ddlEvents";
String ddlCategory = "ddlCategory";
```

İki tane liste kutusu var. Bu kutulardan ilki etkinlik seçiyor. İkincisi ise o etkinliğin barındırdığı koşuları seçiyor.

İlk kutudaki tüm etkinlik bilgilerini veritabanında bir tabloya kaydediyoruz.

Döngü halinde her etkinlik sayfasını selenium driver ile tıklayarak ziyaret ederken;

İkinci kutudaki tüm koşuları da veritabanında diğer bir tabloya kaydediyoruz ve o koşuları

selenium driver ile tıklayıp ziyaret ederken;

Ziyaret edilen sayfa bütün koşu verilerini içerdiğinden sayfanın tamamını Jsoup kütüphanesi ile işlemden geçirip koşuya ait elde ettiğimiz verileri veritabanında ayrı bir tabloya kaydediyoruz.

Benzersiz olan şey, etkinlikler değil, etkinliklerin barındırdığı koşular. Herbirinin benzersiz kimlik numarası var. option elementlerinin value attribute'larından elde ettik onları. Bu koşulara koşu değil de kategori demişler. Yani 5, 10, 42 km gibi olan koşular orada category diye geçiyor.

Dolayısıyla veritabanında kategori ve event ve koşu sonuçları tabloları var toplam 3 tablo.

Etkinlik ve kategori tabloları arasında ilişki, onların benzersiz kimlik numaraları ile sağlandı. Böylece kategorinin hangi etkinliğe ait olduğu anlaşılmış oluyor.

Esas işe yarayan tablo, yani koşuları birbirinden ayıran tablo, kategori tablosu. Fakat, koşu sonuçları tablosunda kategori tablosunun websayfasındaki benzersiz kimlik numarası kullanılırsa, o veri uzun bir metin olduğu için veritabanı dosya boyutunu şişireceğinden, kategori tablosuna fazladan bir sayısal kimlik numarası eklendi ve koşu sonuçları tablosunda yabancı anahtar olarak o kullanıldı.

# Sorular ve Cevaplar

**Websayfalarını indirmek için neden sadece JSoup kütüphanesi değil de, Selenium kullanıldı?**

Çünkü koşu sayfaları her ne kadar tek parça olduğu için indirmesi kolaymış gibi görünse de, sayfanın tamamı yüklenmiyor, kullanıcı sayfayı aşağı kaydırırsa yükleniyor. Bu nasıl anlaşıldı? Jsoup ile indirilen bir sayfada verilerin sayısı 2300 küsür iken, websayfasının sonuna gözle bakıldığında 2800 küsür yani daha fazla olduğu görüldü ve yapay zekaya soru sorulduğunda verdiği cevaptan anlaşıldı. Daha sonra da web gezgini geliştirici araçlarındaki ağ sekmesine bakılarak gözlem yapıldı. Selenium kullanıldığında tüm veri sayfası indirilmektedir. Eğer indirilemeseydi, yine Selenium kullanılarak gerçek insanın websayfasını aşağıya kaydırması ve bir süre beklemesi gibi komutlar koda eklenebilirdi.

**Selenium'un kötü tarafı ne?**

Koddan bağımsız ayrı bir driver olarak işletim sisteminde bulundurulması ve kodda driverin yerinin belirtilmesi gerekiyor. Ayrıca Selenium çalışırken gerçek web gezginini açıyor ve sanki gerçek insan kullanıyormuş gibi işlemler bitene kadar web gezginini açık bırakarak kullanıyor.

**Koda bakıldığında neden her iki liste kutusundaki etkinlik ve kategori verileri döngünün dışında en baştan elde ediliyor?**

Çünkü bir kez elde edip döngü içinde kullanırken bir hata mesajı ile karşılaşılıyor. O sebeple önce ziyaret edilen sayfadaki liste kutularından seçeneklerin idleri elde ediliyro, daha sonra döngü içerisinde DOM elementi olarak o id numarası ile ilgili seçenek seçilip Selenium ile üzerine tıklanıyor.

**Selenium'un işleyiş mantığı Jsoup'dan nasıl farkediyor?**

Selenium ile websayfasını açtıktan sonra, paraşütle atlar gibi sayfaları ziyaret etmiyoruz çünkü ortada sayfalar yok, açtığımız sayfadaki seçeneklere tıklayarak diğer sayfalara ulaşıyoruz. Yani gerçek hayatta web gezginini kullanır gibi kaldığımız bulunduğumuz sayfa üzerinde işlem

yaparak yani birşeylere tıklamasını sağlayarak diğer sayfalara erişiyoruz.

**Programın Kötü Kodlanmış veya Yetersiz Özellikleri neler?**

Program sadece bir kez çalıştırmak üzerine işliyor. Yani başlangıçta veritabanı olmayacak. Veritabanını sıfırdan oluşturacak ve tek çalıştırmada bütün koşu verilerini veritabanına kaydedecek biçimde.

Her category yani koşu verisini elde ettikten sonra veritabanına kaydedip sonrakine geçtiği için en azından bir problem çıksa bile elde biraz veri oluyor.

Fakat benzersiz kimlik numarasını uzun diye kullanmayıp, koşu sonuçları ile kendi verdiğimiz sayısal kimlik numarası ile eşleştirme yaptık ya, ileride websayfasına yeni etkinlik ve koşular eklendiğinde, veya eklenmese bile bizim kod liste kutusundan bilgileri farklı sıra ile çekerse, sayısal kimlik numaraları değiştiğinden aşağıda anlatılacak olan manuel işlemlerden dolayı problem olacaktır.

Özetle program verilerin tek çalıştırmada doğru biçimde elde etmek üzerine kuruludur. Sonradan tekrar çalıştırılması aynı veritabanı üzerinde problem çıkaracaktır. Her defasında eski veritabanını elle silmek gereklidir ki sıfırdan yeni bir veritabanı oluştursun.

**Manuel işlemler nelerdir?**

Websayfasındaki bütün koşuların veritabanına kaydolduğundan emin olmak için mümkünse tüm koşuların sayfaları elle ziyaret edilip sayfaların altına kadar gidip veri sayıları veritabanı ile karşılaştırılmalıdır.

Ayrıca, örneğin kod döngüsündeki son işlemde bir koşunun etkinlik ve category tablolarına veri yazılmış olmasına rağmen koşu sonuçları tablosunda o koşuya ait hiçbir koşu verisi olmadığı görüldü. Bunun sebebi kodlama hatası olabileceği gibi, arka arkaya çok veri indirme işlemi yapıldığından, web sunucusu tarafında bir engelleme veya aksaklık meydana gelmiş olabilir. Bunları denetlemek için bir kodlama yapılmadı.

*Özet*

- Veritabanı programın sadece bir kez çalıştırılmasına dayalıdır ve başlangıçta olmayıp, program tarafından sıfırdan oluşturulması, sonradan ekleme yapılmaması gerekmektedir.
- Websayfasındaki etkinlik, koşu, veri sayıları, veritabanındakilerle gözle karşılaştırılmalı ve eksiksiz olduğundan emin olunmalıdır.

# Kod Kalitesi

- Dökümantasyon olarak bu içerik hazırlandı.
- Test kodları yazılmadı.
- Hata ayıklama konusuna dikkat edilmedi.
- Veritabanı bağlantısı aç kapa işlemlerine dikkat edilmedi.
- Veritabanına veri yazma işlemi performansına yönelik işlem yapılmadı.

- ORM kullanılmadı, düz SQL sorguları PreparedStatement ile birlikte kullanıldı.
- Bu proje sadece veriyi elde etmek için yazıldı. Rapor oluşturma kodları içermemektedir.
- Değişken isimlendirilmelerine dikkat edilmedi. Bazı yerde event id, category id, run id, id gibi değişkenler websayfasındaki isimlendirmelerden dolayı anlam karmaşasına yolaçabilir.

# Veritabanı Yapısı

```
CREATE TABLE categories
(
      id           INTEGER PRIMARY KEY AUTOINCREMENT ①
    , categoryId   TEXT ②
    , categoryName TEXT ③
    , fkEventId    TEXT ④
)

CREATE TABLE events
(
      eventId   TEXT PRIMARY KEY
    , eventName TEXT ⑤
)

CREATE TABLE results
(
      id            INTEGER PRIMARY KEY AUTOINCREMENT
    , ranking       TEXT
    , bibNumber     TEXT
    , fullName      TEXT
    , country       TEXT
    , city          TEXT
    , district      TEXT
    , ageAndGender  TEXT
    , timeAsHHMMSS  TEXT
    , timeAsSeconds INTEGER
    , eventId       INTEGER ⑥
)
```

① results tablosunda ilişkili sütun yer işgal etmesin diye orijinal Idler yerine kendimiz otomatik artan sayısal ID numraları verdik.

② Benzersiz olan ID aslında bu, fakat uzun bir metin olduğundan kullanmadık.

③ Category ismine bakarak koşunun kaç km olduğunu ve normal mi yoksa diğer spor etkinliklerinden(sanal, paten, engelli) biri mi olduğunu anlıyoruz.

④ events tablosundaki Benzersiz olan event ID ile ilişkilidir. Benzersiz koşu kategorisin ait olduğu etkinliğin ismini öğrenmeye yarar.

⑤ Etkinlik ismi etkinliği ismi, hangi yıl nerede yapıldığı gibi bilgilerin bir kısmını içerir.

⑥ Aslında category tablosundaki id sütunundaki değerler ilişkilidir. İsimlendirilmesi yanlış

yapıldı.

`results` tablosundaki TEXT tipi verilerin hepsi internetten elde edildikleri biçimde kaydedildiler.

---

Hazırlayan: Türker Öztürk