GROEIPROJECT
MODULE 8: XML & JSON

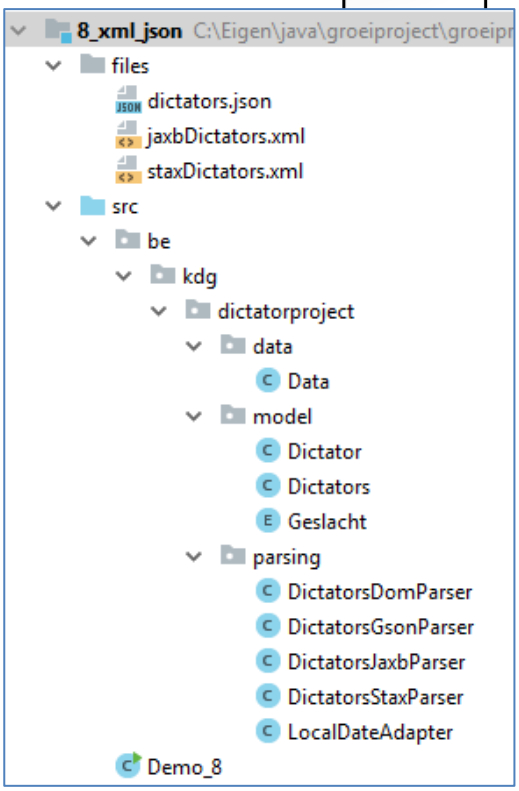
1. Voorbereiding

1.1. Open het groeiproject en maak daarin een achtste module: `8_xml_json`

1.2. Kopieer van module `1_herhaling` de hele src map (met o.a. de oorspronkelijke basis- en multiklasse).

1.3. Zet in de map met de naam `lib` de benodigde jar: `gson-2.4.` Je vindt die op Blackboard. Gebruik de menukeuze "*File* > *Project Properties* > *Project Settings* > *Libraries*" om naar deze jar te verwijzen.

1.4. Maak in de module `7_xml_json` een nieuwe directory (dus **NAAST** `src`) met de naam `files`. Hierin komen de databestanden te staan.
Hiernaast een screenshot van de gewenste projectstructuur als alles klaar is →



2. XML

2.1. We gaan het parsen naar XML eerst uitproberen met **StAX** technologie.
Maak een nieuwe package `be.kdg.`*`projectnaam`*`.parsing` en daarin een klasse met de naam `XxxStaxParser` (in ons voorbeeld: `DictatorsStaxParser`).

2.2. Voorzie een constructor met 2 parameters: de multiklasse en het pad waarnaar we de data gaan wegschrijven (zie 1.3). In de constructor maak je de `XMLStreamWriter` klaar voor gebruik.

2.3. Voorzie een methode `writeXML()` die het volledige multiobject met StAX wegschrijft.

a) Organiseer je code: schrijf elk element weg via een afzonderlijke methode; bijvoorbeeld:

```
private void writeElement(Dictator dictator)
```

b) Maak gebruik van de **StAX**-technologie om alle eigenschappen van elk basisobject weg te schrijven, voorzien van de juiste start- en end tag. Eén van de eigenschappen van het basisobject gebruik je als **attribute** in de start tag van dat element (In ons voorbeeld is dat het regime dat de dictator gevoerd heeft)

c) Denk er ook aan dat je alle waarden moet omzetten naar String (dus ook enum, LocalDate, int, double…)

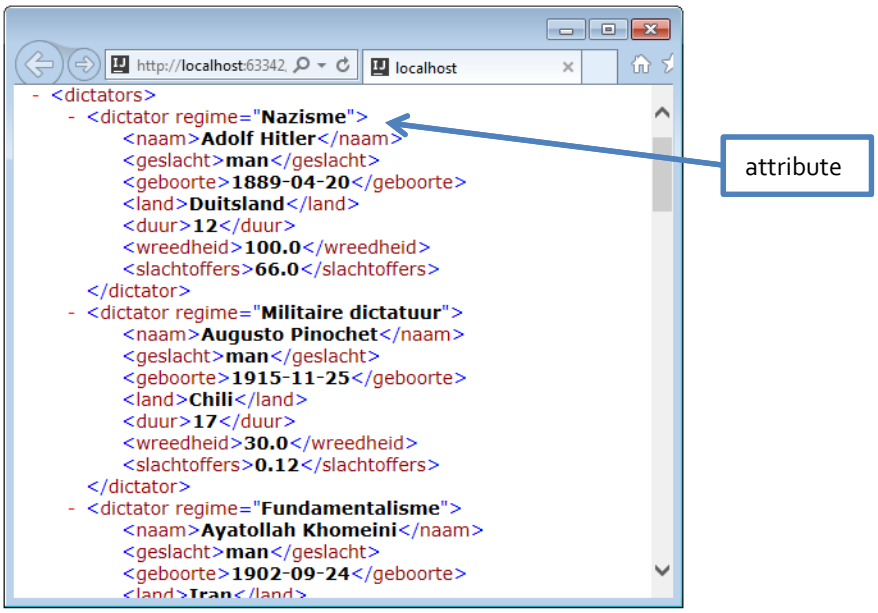
2.4. Maak een klasse `Demo_8` met een `main`

a) Vraag aan de klasse `Data` een gevulde List met basisobjecten en maak daarmee een multiobject aan.

b) Creëer de `XxxStaxParser` en roep de methode `writeXml` op.

c) Controleer de inhoud van de XML-file (zie volgende pagina)

2.5. Controleer de inhoud van het xml-bestand (bijvoorbeeld met een webbrowser):



Het gemaakte XML bestand is perfect om over het netwerk te zenden of voor een computer, maar niet zo goed leesbaar. Maak het bestand leesbaarder door het weg te schrijven met een pretty printing streamwriter die je maakt op basis van de `XMLStreamWriter`:

```
myStreamWriter = new IndentingXMLStreamWriter(myStreamWriter);
```

Let op: de `IndentingXmlStreamWriter` zit in de package com.sun.xml.internal.txw2.output. De klasse zit in de JDK, maar maakt geen deel uit van de standaard. Vanaf Java 9 is ze met het nieuwe module systeem niet meer toegangkelijk.

2.6. Maak in package parsing een nieuwe klasse `XxxDomParser` om het xml-bestand met DOM-technologie om te zetten naar bruikbare data. In ons voorbeeld is dat: `DictatorsDomParser`.
Voorzie de volgende methode:

```
public static Dictators jdomReadXml(String fileName)
```

a) Maak gebruik van **DOM**-technologie om het xml-bestand element per element in te lezen.

b) Opgelet: alle waarden komen als string binnen en moet je dus nog op gepaste wijze omzetten naar enum, LocalDate, int, double, ...
Voor LocalDate kan dat met de methode: `LocalDate.parse(string);`

c) De returnwaarde is een gevuld multiobject (in ons voorbeeld: `Dictators`)

d) Maak voor het lezen van eenvoudige tekst elementen in deze klasse een static helper methode. Deze neemt als parameters het parent element en de naam van het gezochte child element. De methode geeft de inhoud van het child terug:

```
private static String getChildText(Element parent, String name)
```

2.7. In de klasse `Demo_8` ga je deze methode uittesten. Druk alle waarden af en controleer:

```
Na wegschrijven via STAX /inlezen via DOM:

Adolf Hitler          (°1889) Duitsland       regime: Nazisme              66,0 mln doden
Augusto Pinochet      (°1915) Chili           regime: Militaire dictatuur   0,1 mln doden
Ayatollah Khomeini    (°1902) Iran            regime: Fundamentalisme       8,5 mln doden
Benito Mussolini      (°1883) Italië          regime: Fascisme              2,0 mln doden
enz...
```

2.8. We gaan nu automatische binding toepassen via **JAXB**-technologie. Zorg dat de inhoud van het XML bestand dat door JAXB geschreven wordt dezelfde is als het XML document dat je met StAX schreef. Daartoe moeten we eerst een aantal aanpassingen doen aan onze basisklasse en onze multiklasse:

a) In de **basisklasse** (in ons geval: `Dictator`):
- Gebruik een annotatie om de volgorde van de tags vast te leggen
- Gebruik een annotatie boven een van de setters waar je naar een XML attribuut moet mappen
- Indien er attributen zijn met hoofdletters in het midden, annoteer de methoden dan om ze weg te schrijven volgens de XML conventie (kleine letters, met een koppelteken voor de plaats waar de hoofdletter stond)
- Gebruik `@XmlJavaTypeAdapter` boven elke setter van het type LocalDate. Verwijs daar naar een apart te schrijven klasse `LocalDateAdapter` (zie voorbeeld op BB)
- Controleer of er een default constructor aanwezig is (wordt opgroepen door JAXB)

b) In de **multiklasse** (in ons geval: `Dictators`):
- Gebruik `@XmlRootElement`
- Verander de ingekapselde collectie naar `ArrayList` en voorzie een setter en getter naar deze list (in ons voorbeeld: `setDictatorList` en `getDictatorList`). JAXB zal voor ons deze list automatisch vullen.
- Gebruik `@XmlElement` boven de methode `setDictatorList`

2.9. Maak een nieuwe klasse `XxxJaxbParser` (in ons voorbeeld `DictatorsJaxbParser`).
Voorzie daar volgende static methoden:

a) `public static void JaxbWriteXml(String file, Object root)`
Maak gebruik van een **Marshaller** om de data te converteren naar een nieuw XML-bestand.

b) `public static <T> T JaxbReadXml(String file, Class<T> typeParameterClass)`
Maak de methode generiek voor om het even welke multiklasse.
Maak gebruik van een **Unmarshaller** om de data te lezen uit een XML-bestand.

2.10. Voeg een test toe in `Demo_8` om het resultaat te controleren:

```
Na wegschrijven/inlezen via JAXB:

Adolf Hitler           (°1889) Duitsland       regime: Nazisme               66,0 mln doden
Augusto Pinochet       (°1915) Chili           regime: Militaire dictatuur    0,1 mln doden
Ayatollah Khomeini     (°1902) Iran            regime: Fundamentalisme        8,5 mln doden
Benito Mussolini       (°1883) Italië          regime: Fascisme               2,0 mln doden
enz...
```

3. JSON

3.1. Maak een nieuwe klasse `XxxGsonParser` (in ons voorbeeld `DictatorsGsonParser`) en daarin een klassemethode om de data in JSON-formaat weg te schrijven via **Gson**-technologie.
In ons voorbeeld wordt dat:

```
public static void writeJson(Dictators dictators, String fileName)
```

3.2. Schrijf ook een methode om de data in JSON-formaat in te lezen via **Gson**-technologie.
In ons voorbeeld wordt dat:

```
public static Dictators readJson(String fileName)
```

3.3. Voeg een test toe in `Demo_8` en verifieer de JSON-array in het geschreven bestand. Links zie je een voorbeeld van een oplossing met het wegschrijven van een lijst van basisobjecten, rechts een oplossing met het wegschrijven van het multiobject:

```
[
  {
    "naam": "Ayatollah Khomeini",
    "geslacht": "MAN",
    "geboorte": {
      "year": 1902,
      "month": 9,
      "day": 24
    },
    "land": "Iran",
    "regime": "Fundamentalisme",
    "duur": 10,
    "wreedheid": 99.7,
    "slachtoffers": 8.5
  },
  {
    "naam": "Nicolae Ceausescu",
    "geslacht": "MAN",
    "geboorte": {
      "year": 1918,
      "month": 1,
      "day": 26
    },
    "land": "Roemenië",
    "regime": "Despotisme",
    "duur": 24,
    "wreedheid": 80.0,
    "slachtoffers": 0.5
  },
  {
```

```
{
  "dictators": [
    {
      "naam": "Ayatollah Khomeini",
      "geslacht": "MAN",
      "geboorte": {
        "year": 1902,
        "month": 9,
        "day": 24
      },
      "land": "Iran",
      "regime": "Fundamentalisme",
      "duur": 10,
      "wreedheid": 99.7,
      "slachtoffers": 8.5
    },
    {
      "naam": "Nicolae Ceausescu",
      "geslacht": "MAN",
      "geboorte": {
        "year": 1918,
        "month": 1,
        "day": 26
      },
      "land": "Roemenië",
      "regime": "Despotisme",
      "duur": 24,
      "wreedheid": 80.0,
      "slachtoffers": 0.5
    },
```

3.4. Controleer ook de output van `Demo_8`:

```
Na wegschrijven/inlezen via GSON:

Adolf Hitler           (°1889) Duitsland      regime: Nazisme               66,0 mln doden
Augusto Pinochet       (°1915) Chili          regime: Militaire dictatuur    0,1 mln doden
Ayatollah Khomeini     (°1902) Iran           regime: Fundamentalisme        8,5 mln doden
Benito Mussolini       (°1883) Italië         regime: Fascisme               2,0 mln doden
enz...
```