Formális Nyelvek

# ANTLR

Bressel Márk

Horváth Ákos

Bereczki-Király Rómeo

# Mi az az ANTLR?

Az **ANTLR** (Another Tool for Language Recognition) egy erőteljes és rugalmas eszköz, amelyet programozási nyelvek, konfigurációs fájlok, adatfolyamok és más szöveges formátumok elemzésére használnak. Az ANTLR segítségével könnyedén definiálhatók egy nyelv szintaktikai szabályai, majd ezekből automatikusan generálható parszolók (parse-ek) különböző programozási nyelveken, például Python, Java, C#, stb. A generált parszolók képesek felismerni a nyelv szintaktikai szerkezetét, és feldolgozni a bemenetet az adott nyelv szabályai szerint.

# ANTLR használata a projektünkben

A projektben az ANTLR-t arra használtuk, hogy egy JSON elemzőt hozzunk létre. A JSON egy könnyen olvasható adatcsere formátum, amelyet széles körben használnak különböző alkalmazásokban. Az ANTLR segítségével definiáltuk a JSON szintaxisát, majd generáltuk a szükséges parszolókódot, amely képes a JSON fájlok elemzésére és feldolgozására.

# Projekt felépítése/Fontosabb fileok

## JSON.g4 - A JSON Nyelvtan

> A JSON.g4 fájl az ANTLR által használt nyelvtani definíciókat tartalmazza. Mi ezt a fájlt használtuk a JSON formátum szintaktikai szabályainak meghatározására, amelyek alapján az ANTLR képes generálni a lexer és parser kódot.

**Alapvető Szabályok:**

- **json:** A JSON szerkezet belépési pontja, amely egy értéket (value) követően várja a fájl végét (EOF).
- **obj:** Egy JSON objektum, amely kulcs-érték párokat tartalmazhat, vagy üres lehet.
- **arr:** Egy JSON tömb, amely értékek listáját tartalmazhat, vagy üres lehet.
- **value:** Egy JSON érték, amely lehet string, szám, objektum, tömb, true, false, vagy null.

**Tokenek és Fragmentek:** A nyelvtan definiálja a JSON alapvető tokenjeit, például a STRING, NUMBER és WS (whitespace) tokeneket, valamint a karakterláncokhoz, számokhoz és escape karakterekhez szükséges fragmenteket.

A JSON.g4 a JSON formátum teljes szabályrendszerét lefedi, amely alapján az ANTLR létrehozza az elemző program alapját.

## JSON.interp - A Nyelvtan Értelmezése

> Ez az automatikusan generált fájl tartalmazza a JSON.g4 alapján létrejött nyelvtani értelmezést, amely az ANTLR belső működését támogatja.

**Tokenek:** Listázza a tokeneket szó szerinti (literal names) és szimbolikus (symbolic names) neveikkel együtt, például {, }, vagy STRING.

**Szabályok:** A JSON-nyelvtan fő szabályai, mint például json, obj, pair, arr, és value, itt is szerepelnek.

**ATN:** Egy belső reprezentáció az elemző automatájáról (ATN - Abstract Transition Network), amely a JSON bemenet feldolgozását végzi.

Ez a fájl biztosítja a lexer és parser működéséhez szükséges információkat.

## JSON.tokens - Tokenek Leírása

Ez a fájl tartalmazza az ANTLR által generált tokenek listáját, amelyek a JSON szintaktikai elemeit azonosítják, például az alábbiakat:

- **STRING** (karakterlánc),
- **NUMBER** (szám),
- **T__0–T__8** (szimbólumok, mint {, }, :, [, ]).

Ez a fájl kulcsszerepet játszik abban, hogy az ANTLR felismerje a JSON szintaktikai elemeit, és segít a lexer működésében.

## JSONLexer.interp - Lexer Specifikáció

Ez az automatikusan generált fájl tartalmazza a lexer (szóelemző) implementációjához szükséges információkat.
**Token Definíciók:** Tartalmazza a tokenek literális és szimbolikus neveit, például STRING, NUMBER, és WS.

**ATN:** Egy állapotgép (state machine) reprezentációja, amely segít a lexernek feldolgozni a bemenetet.

Ez a fájl alapot ad a lexer pontos működéséhez.

## JSONLexer.py - Lexer Implementáció

A JSONLexer.py az ANTLR által generált lexer kódot tartalmazza, amely a JSON bemenetet tokenekre bontja.

**Lexer Funkciói:**

- Beolvassa a bemeneti szöveget.

- Felismeri a JSON elemeit (pl. stringek, számok, whitespace karakterek).
- Tokenekre bontja a bemenetet.

**ATN és DFA:** Tartalmazza az állapotgépet (ATN) és a determinisztikus véges automatát (DFA), amelyek segítik az elemzést.

Ez a lexer felelős a JSON szintaktikai elemeinek felismeréséért, amelyeket a parser dolgoz fel.

## JSONLexer.tokens - Lexer Tokenek

> Ez a fájl a lexer által használt tokeneket írja le, amelyek megegyeznek a JSON.tokens tartalmával. A tokenek numerikus azonosítókat és azok szó szerinti reprezentációit is tartalmazzák.

Ez a fájl segíti a lexer működését, hogy pontosan felismerje a JSON elemeit.

## JSONListener.py - Parse Tree Listener

> A JSONListener.py egy automatikusan generált osztályt tartalmaz, amely lehetővé teszi az eseményalapú feldolgozást a parse tree (feldolgozási fa) csomópontjain.

**Enter és Exit Metódusok:** Minden parse tree szabályhoz tartozik egy enterRuleName és exitRuleName metódus, amelyek akkor futnak le, amikor a parser belép vagy kilép egy adott szabály csomópontjából.

**Alapértelmezett Implementáció:** A metódusok alapértelmezés szerint üresek, de testreszabhatók a további feldolgozási igények szerint.

Ez a fájl lehetőséget biztosít a parse tree csomópontjaihoz kapcsolódó logika hozzáadására.

## JSONParser.py - Parser Implementáció

A JSONParser.py az ANTLR által generált parser kódot tartalmazza, amely a lexer által létrehozott tokenekből épít feldolgozási fát.

**Szabályok és Kontextusok:**

- **Szabályok:** json, obj, pair, arr, value.
- **Kontextusok:** Minden szabályhoz kapcsolódik egy kontextus osztály, amely az adott szintaktikai elemhez tartozó adatokat tárolja.

**Parse Tree Létrehozása:** A parser a tokenek alapján hierarchikus szerkezetet hoz létre, amely a JSON bemenet szerkezetét tükrözi.

Ez a parser a JSON bemenet feldolgozásának alapvető eleme.

## JSONParser.py - Kontextusok és Feldolgozás

A JSONParser.py kiterjed a parse tree létrehozására és a különböző JSON elemek kezelésére.

**Metódusok:**

- **json():** A teljes JSON elemzése.
- **obj():** Egy JSON objektum kezelése.
- **pair():** Kulcs-érték párok feldolgozása.
- **arr():** JSON tömbök kezelése.
- **value():** Általános JSON értékek feldolgozása.

Ez a fájl biztosítja a parse tree pontos strukturális feldolgozását.

## JSONVisitor.py - Parse Tree Látogató

A JSONVisitor.py egy visitor osztályt tartalmaz, amely lehetővé teszi a parse tree bejárását és az adatok feldolgozását.

**Visit Metódusok:** Minden szabályhoz tartozik egy visitRuleName metódus, amely testreszabható feldolgozást végezhet az adott csomóponton.

Ez a fájl a parse tree feldolgozásának rugalmas eszköze.

## tests.json - Teszt Adatfájl

Ez a JSON fájl különböző adatokat tartalmaz, amelyek lefedik a JSON formátum minden alapvető elemét:

- Kulcs-érték párok,
- Beágyazott objektumok,
- Tömbök,
- Boolean, null és string típusok.

Ez a fájl a feldolgozó eszköz működésének validálására szolgál.

## Main.py - Fő Program és PrettyPrinter

A Main.py a JSON elemzés teljes folyamatát összefogja.

**Lexer és Parser:** Beolvassa a bemenetet, és létrehozza a parse tree-t.

**JSONPrettyPrinter:** Egy egyedi visitor osztály, amely bejárja a parse tree-t, és Python adatstruktúrába konvertálja a JSON adatokat.

**Kimenet:** A parse tree feldolgozása után szépen formázott JSON-t nyomtat ki.

Ez a fájl a projekt végrehajtható fő programja.