Najpopularniejsze biblioteki wykorzystane w projekcie:

Json-simple

JSON to prosty format wymiany danych. Łatwo go odczytują komputery, oraz jest czytelny dla ludzi. Jego definicja opiera się o podzbiór języka programowania JavaScript, Standard ECMA-262 3rd Edition - December 1999.

JSON jest formatem tekstowym, stosowanym w językach takich jak C, w tym C++, C#, Java, JavaScript, Perl, Python i wielu innych. Dzięki temu nadaje się on doskonale jako pośrednik między różnymi środowiskami oprogramowania.

JSON powstał w oparciu o dwie struktury:

- Zbiór par nazwa/wartość. W różnych językach jest to implementowane jako obiekt, rekord, struktura, słownik, tabela hash, lista z kluczem, albo tabela asocjacyjna.

- Uporządkowana lista wartości. W większości języków implementuje się to za pomocą tabeli, wektora, listy, lub sekwencji.

Wspomniane struktury danych są uniwersalne. Prawie wszystkie nowoczesne języki programowania posługują się nim w tej lub innej formie. Ma to sens, by format danych, który jest przenośny pomiędzy różnymi językami programowania opierał swoją budowę na wspomnianych strukturach.

W przypadku formatu JSON, przybierają one następujące formy:

Obiekt jest nieuporządkowanym zbiorem par nazwa/wartość. Opis obiektu rozpoczyna { (lewa klamra) a kończy } (prawa klamra). Po każdej nazwie następuje : (dwukropek) oraz pary nazwa/wartość, oddzielone , (przecinkiem).

Tabela jest uporządkowanym zbiorem wartości. Opis tabeli rozpoczyna znak [(lewy nawias kwadratowy) a kończy znak] (prawy nawias kwadratowy). Poszczególne wartości rozdialane są znakiem , (przecinek).

Wartość to łańcuch znakowy, którego początek i koniec oznacza podwójny cudzysłów, lub liczba, lub wartość true (prawda) lub false (fałsz) lub null, lub obiekt lub tabela. Struktury te można zagnieżdżać.

Łancuch znakowy jest zbiorem zera lub większej ilości znaków Unicode, opakowanym w podwójne cudzysłowy, stosujących znak odwrotnego ukośnika jako początek sekwencji specjalnej (escape). Pojedynczy znak jest reprezentowany

Mysql connector

```
}
String url = "jdbc:mys.ql://db";
try {
  connection = DriverManager.getConnection(url, "user", "password");
} catch (SQLException ex) {
  LOG.log(Level.SEVERE, "Couldn't connect with database", ex);
  return;
}
LOG.log(Level.INFO, "Connected with database");
```

MySQL zapewnia możliwość łączności z bazą danych MySQL w języku Java za pomocą biblioteki MySQL Connector/J, a MySQL Connector/J jest czwartym typem sterownika JDBC. Ten typ oznacza czystą implementacje protokołu MySQL.

W przypadku programów na dużą skalę, które używają wspólnych wzorców projektowych dostępu do danych, zaleca się rozważyć użycie jednego z popularnych frameworków takich jak Hibernate, Spring JDBC lub iBatis SQL Maps, aby zmniejszyć ilość kodu JDBC do debugowania, dostrojenia, zabezpieczenia i utrzymania.

File Handler

```
LOG.setLevel(Level.WARNING); // Level of logs to write (default: WARNING)
fileTxt = new FileHandler("Log.txt");

// create a TXT formatter
SimpleFormatter formatterTxt = new SimpleFormatter();
fileTxt.setFormatter(formatterTxt);
LOG.addHandler(fileTxt);
}
```

The FileHandler jest w stanie zapisywać dane do pliku, jak również do zestawu plików.

W podanym zestawie plików, gdy w jednym zostanie osiągnięty podany limit rozmiaru, uzupełnianie go zostaje zamknięte i otwiera się kolejny plik. Kolejno do starszych plików dodaje się 0, 1, 2… itd.

Domyślnie klasa XMLFormatter jest używana w formatowanym tekście.

Ciąg znaków odwołujący się do ścieżki pliku może zawierać następujące specjalne komponenty:

- "/" lokalna ścieżka separatora
- "%t" tymczasowy katalog systemowy
- "%h katalog domowy użytkownika
- "%u" numer unikalny w przypadku konfilktów
- "%%" przekład na pojedynczy znak procenta (%)

String builder

```
this.list = list;
pw = new PrintWriter(new File("plan.txt"));
sb = new StringBuilder();

// Header (optional)
//sb.append("course,room,professor,day,start_hour,90,group_name");
```

Zmienny ciąg znaków. Klasa ta udostępnia API kompatybilnego z StringBuffer, ale nie gwarantuje synchronizacji. Klasa ta jest przeznaczona do stosowania jako zamiennik drop-in StringBuffer w miejscach, gdzie łańcuch bufora był używany przez jeden wątek.

Podstawowymi operacjami string buildera są appena oraz przeciążone metody umieszczania, dla dowolnych typów danych. Każdy skutecznie przekształca dany punkt odniesienia do łańcucha, a następnie dodaje lub wstawia znaki tego łańcucha do konstruktora ciągu stringa.

Metoda append zawsze dodaje te znaki na końcu konstruktora; metoda umieszczania dodaje znaki w określonym punkcie.

Na przykład, jeśli Z odnosi się do obiektu ciąg konstruktora, którego zawartość to "Start", a następnie użyjemy append ("le") spowodowałoby to, że ciąg będzie zawierał "startle", natomiast z.insert (4 " le ") zmieniłoby to wartość na "starlet".

Użycie StringBuilder nie jest bezpieczne przy wielu wątkach. W przypadku konieczności synchronizacji danych między wątkami, zaleca się StringBuffer.

Bibliografia:

http://www.json.org/json-pl.html

https://docs.oracle.com/javase/7/docs/api/java/util/logging/FileHandler.html

https://dev.mysql.com/doc/connector-j/5.1/en/connector-j-overview.html

https://docs.oracle.com/javase/7/docs/api/java/lang/StringBuilder.html