# Format CSV- specyfikacja

Nie istnieje żadna formalna specyfikacja formatu csv, co pozwala na szeroką implementację tego formatu. Można jednak wyróżnić pewne założenia, które są stosowane w większości implementacji:

1. Każdy rekord znajduje się w osobnej linii, które wyznacza znak CRL, jednak końcowy rekord nie zawsze musi kończyć się tym znakiem.
2. Plik csv może posiadać opcjonalny nagłówek, który określna nazwy pól rekordów. Musi znajdować się on w pierwszej linii oraz posiadać tyle samo pól, co reszta rekordów.
3. Wszystkie rekordy pliku powinny posiadać tyle samo pól, które są rozdzielone przecinkiem lub średnikiem.
4. Spacje są częścią każdego pola, nie powinny być ignorowane.
5. Każde pole może znajdować się w podwójnym cudzysłowie, które powinny być ignorowane. Nie ma wymogu konsekwentnego ich stosowania w całym pliku.
6. Pola, które zawierają znak nowej linii powinny znajdować się w podwójnym cudzysłowie.
7. Jeśli pole zawiera cudzysłów, powinien być on zastąpiony podwójnym.

# Istniejące rozwiązania- biblioteki Java

Istnieje wiele bibliotek do obsługi plików csv napisanych na platformę Java. Większość z nich jest tylko prostymi parserami. Jednakże, ze względu na ich ilość, wybór tych, które wspierają mapowanie obiektowe, wciąż jest duży. Poniżej przedstawionych jest kilka popularnych bibliotek, wraz z krótko przedstawionymi kluczowymi funkcjonalnościami.

1. Apache OpenCSV- najpodatniejsza, najbardziej rozbudowana biblioteka, będąca wzorem dla opracowywanego rozwiązania.
   * wspieranie POJO, kolekcji, typów wbudowanych,
   * mapowanie pól na podstawie nazwy i kolejności,
   * obsługa tablic przy pomocy adnotacji,
   * obsługa niestandardowej konwersji przy pomocy adnotacji,
   * obsługa relacji one-to-one, one-to-many i many-to-one(MultiValueMap)
   * obsługa relacji rekurencyjnych,
   * dodawanie walidatorów pojedynczych pól i całych rekordów.
2. SuperCSV
   * wspieranie POJO, map, list, typów wbudowanych,
   * automatyczne wykrywanie kodowania,
   * dodawanie własnych konwerterów dla obiektów,
   * dodawanie walidatorów,
   * częściowe czytanie/zapisywanie plików.
3. UniVocity CSV Parser
   * Obsługa sterowania mapowaniem przy pomocy adnotacji oraz tzw Fluent API.
   * Szeroki zestaw adnotacji dla przetwarzania wartości,
   * dodawanie własnych konwerterów,
   * częściowe zapisywanie/odczytywanie,
   * wygodna obsługa błędów.
4. FlatPack- bardzo prosta biblioteka, brak mapowania wprost, w zasadzie parser z możliwością dodawania konwerterów.

# Wnioski z analizy

* Istnieje dużo bibliotek do obsługi plików csv, jednak, w porównaniu z innymi formatami, nie jest ich aż tak wiele. Wpływ na to ma, moim zdaniem, brak sprecyzowanego standardu oraz prostota formatu.
* Większość bibliotek, do sterowania procesem mapowania stosuje adnotacje.
* Biblioteki pozwalają na dodawanie własnych konwerterów oraz na walidację obiektów (również przy pomocy własnych walidatorów).
* Ze względu na brak jasnej specyfikacji nie istnieje jeden sposób na odwzorowywanie relacji między obiektami. Najczęstszym podejściem jest serializowanie obiektów do postaci tekstowej i umieszczanie ich jako pola rekordu. Drugim, rzadziej stosowanym podejściem jest rozdzielenie relacji do innych plików- analogicznie do tabel relacyjnych baz danych.
* Najczęściej mapowaną relacją (jeśli są one w ogóle mapowane) jest one-to-one. Rzadziej implementowane są relacje one-to-many i many-to-one. Relacja many-to-many jest implementowana rzadko i nie znajduje się wprost w żadnej popularnej bibliotece.

# Bibliografia

1. <https://tools.ietf.org/html/rfc4180>
2. <https://stackabuse.com/libraries-for-reading-and-writing-csvs-in-java/>
3. [https://www.univocity.com/](https://www.univocity.com/pages/about-parsers)
4. <http://flatpack.sourceforge.net/>
5. <https://super-csv.github.io/>
6. <http://opencsv.sourceforge.net/>