Web スクレイピング

Pythonによる インターネット情報活用術

豊沢 聡◉著

■**サンプルファイルのダウンロードについて**

本書掲載のサンプルファイルは、下記 URL からダウンロードできます。

https://----------

・本書の内容についてのご意見、ご質問は、お名前、ご連絡先を明記のうえ、小社出版部宛文書（郵送または E-mail）でお送りください。
・電話によるお問い合わせはお受けできません。
・本書の解説範囲を越える内容のご質問や、本書の内容と無関係なご質問にはお答えできません。
・匿名のフリーメールアドレスからのお問い合わせには返信しかねます。

はじめに

本書では、Python を用いた Web スクレイピングの方法を示します。

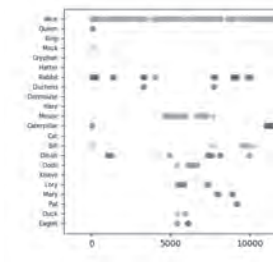
Web スクレイピングは、ネット上のさまざまな情報を取り込み、必要なものだけを抽出し、まとめを提示する技術です。総称であって、単一の技術ではありません。ネットに散在するデータの種類と形式、得たい情報と提示方法の多様さを考えればわかるように、1 つの方法ですべてをカバーすることなどできないからです。本書では、目的とサイトのデータにあわせていろいろなスクリプティングの方法を説明します。

本書で紹介するトピックを次に示します。左側の画像はそれぞれの出力結果です。

第 2 章 TXT 英語

物語（英語テキスト）をダウンロードし、人名および地名を重要度（出現頻度）に応じた大きさの文字で画像に張り付けます。ここから、主要な登場人物が一瞥で把握できます。例題に用いるデータソースはプロジェクト・グーテンベルグ掲載の、ホメロス作『イリアス』です。使用する Python ライブラリは、英文処理では NLTK（Natural Language Tool Kit）、画像生成では WordCloud です。

第 3 章 TXT 英語



物語（英語テキスト）をダウンロードし、キャラクターの登場箇所をプロットすることで、ストーリーラインを示します。登場人物の出番がわかるので、物語の進行がおおざっぱに理解できます。例題はこちらもプロジェクト・グーテンベルグ掲載で、ルイス・キャロル作『不思議の国のアリス』です。英文処理には前章と同じく NLTK を用いますが、画像生成には Matplotlib を使います。

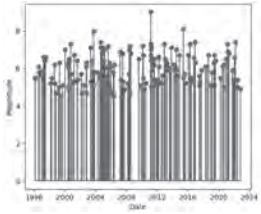
第 4 章 HTML 日本語

商用サイトの HTML ページ（日本語）を取得し、一般名詞と固有名詞の大きさを重要度に応じて変えて画像に張り付けます。ここでは本書出版社の出版目録をターゲットとすることで、出版元の出版傾向を一覧します。手法は第 2 章と同じですが、ターゲットが日本語 HTML になったことで、テキスト処理ライブラリが Beautiful Soup と Janome に変わります。

第 5 章 ZIP TXT 日本語

物語（日本語テキスト）をダウンロードし、人名および地名の大きさを重要度に応じて画面に張り付けます。例題は青空文庫掲載の、太宰治作『人間失格』です。第 2 章、第 4 章と同じパターンですが、データソースが Zip 化されているので、それに対処する処理が加わります。

第 6 章 HTML 表 日本語



HTML ページ（日本語）に掲載された表を抽出し、グラフを描きます。例題では気象庁の地震データを取り上げ、地震の時系列的な頻度グラフと強度のヒストグラムを示します。使用ライブラリは表解析の Pandas とグラフプロットの Matplotlib です。

第 7 章 HTML 画像 Pickle

HTML ページに掲載された画像をすべて抜き出し、保存（Pickle 化）します。いったんローカルに保存するのは、続く 3 つの章で同じ画像ファイルを使いまわすからです。これで、サーバやネットワークに余計な負荷を掛けなくて済みます。ターゲットソースはどこでも構いませんが、本書ではこの本の出版社を使います。HTML からの画像リンク抽出には第 4 章と同じ Beautiful Soup を、画像オブジェクトデータの保存には Python 標準ライブラリの Pickle（漬物の意）を使います。

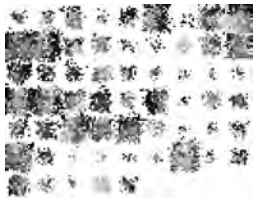
第 8 章 Pickle 画像

前章の Pickle ファイルからアニメーション画像を作成します（紙面では動きませんが）。ページ内に散らばる画像を素早くチェックできるようにするのが目的です。画像処理のライブラリは Pillow です。

第 9 章 Pickle 画像

Pickle ファイルからサムネール画像を作成します。これも目的は前章と同じく一覧性です。画像処理のライブラリも同じ Pillow です。

第 10 章 Pickle 顔画像



HTML ページに掲載された画像のうち、人物の顔の部分だけを抜き出してサムネール画像を生成します。ニュースサイトなど人物写真の多いページに便利です。手法的には、第 9 章に顔検出機能を加えたもので、より高度な画像処理のできるコンピュータビジョンライブラリの OpenCV を使います。

第 11 章 JSON 地図

JSON 形式にフォーマットされた地理座標を取得し、地図にマーキングします。地図はスクロールや拡大縮小のできるインタラクティブマップです（JavaScript コードが生成されます）。例題は東京都が提供する都内の無料 Wi-Fi スポットデータで、アクセス方法は REST API です。地図生成には Plotly を使います。

第 12 章 CSV 地図 日本語

CSV 形式の表から地理座標を取得し、インタラクティブマップにマーキングします。例題は国土交通省の駅データです（すべての駅が網羅されているわけではありません）。地図生成は第 11 章と同じなので使用するのも Plotly ですが、駅名を加えたり、駅の規模に応じてマーカーサイズを変更するなどの改良が施されています。複数（JR、私鉄、地下鉄）の CSV 表を組み合わせるなどの表操作には Pandas を使います。

紙面では動かないアニメーション画像やインタラクティブマップは、次に示す筆者の Github ページから確認してください。

```
https://github.com/stoyosawa/ScrapingBook-Public
```

スクリプティングのベース言語は Python です。上記に示した各種の外部パッケージは、それぞれそれ自体が 1 冊の書籍でもカバーしきれないほどの機能があるので、本書で紹介するのはごく一部です。もっとよい、もしくは効率的な方法を知りたい、あるいは違ったデータや表現を扱いたいという読者は、それぞれの書籍あるいはオリジナルのリファレンスマニュアルを参照してください。大半が英語なので最初は戸惑いますが、本書で取っ掛かりが得られたあとなら、それほど苦には感じないと思います。

ネットは膨大な量の情報で満ちています。あれこれ探索して活用していただければ幸いです。

2023 年 7 月
豊沢 聡

注意事項

以下、本書で注意すべき点を説明します。

■ 実行環境

Python はプラットフォーム非依存なので OS は問いませんが、Windows あるいは Windows Subsystem for Linux（WSL）での実行を念頭に説明しています。したがって、用例のプロンプトマークは `C:>temp` または `$` です。個々のメソッドの用法は Python のインタラクティブモードから示します（プロンプトは >>>）。

スクリプトは生成画像をローカルに保存します。手持ちの画像ビューワーから閲覧してください。一部、スクリプトから直接表示するものもありますが、ディスプレイのない仮想マシンでは画像は表示できません。それらのスクリプトはホストマシンで実行してください。

Google Colab などのオンライン環境で実行するときは、その環境の用法を参照してください。本書ではオンライン環境は説明しません。

■ サンプルスクリプト

本書のサンプルスクリプトは、出版社のダウンロードサービスからダウンロードできます。ダウンロードサービスには、リンク付きの参考文献やサンプル出力画像も含まれています。

サンプルスクリプトは目的を達成できる最小限で書かれています。例外にはほとんど対処しないので、エラー終了することもある点、ご了承ください。

本書のスクリプトは、比較的汎用性のあるものもあれば、特定のデータソースでなければ動作しないものもあります。画像関係は中身の解釈に立ち入らないので、適用性は高いです。自然言語関係は品詞を使って解析するので、やや応用が利きます。ただし、性質の異なるデータソースでは、思ってもいない結果が出ることもあります。表関係は列名を直接指定するので、変更なしでは例題の表でしか使えません。その代わり、意図通りの結果が得られます。

本書の目的は、即座に利用できるスクリプトを提供することではなく、スクレイピングのいろいろな方法を概略的に示すところにあります。アプローチの仕方がわかったら改変する、他と組み合わせるなど、いろいろなパターンを試してください。本文でカバーしていない事例については、付録 A にまとめたので参考にしてください。

Python について

使用するのは Python 3 です。とくに凝った用法は用いていないので、マイナーバージョンは問いません。しかし、最新でないならこれを機会にアップデートするとよいでしょう。

本書は言語としての Python そのものの指南書ではないので、一般的な用法は説明しません。たとえば、リストおよび辞書内包表記、f-string、str の各種メソッドなどスタンダードな機能は説明なしで使っています。set のようなデータ型、正規表現、Zipfile、pickle 等のあまり使わないモジュールは要所で説明していますが、本書で使う範囲内だけです。細かい点は、Python のリファレンスを参照してください。

本書で用いる標準ライブラリは concurrent、io、math、pickle、random、re、statistics、sys、timeit、urllib、zipfile です。

外部パッケージのインストール

Python 標準ライブラリに含まれていない外部のパッケージ（ライブラリ）は、インストールが必要です。それぞれの章でその都度インストール方法は説明していますが、必要なパッケージをすべて一気にインストールするのなら、次を実行してください。

```
python -m pip install --upgrade pip
pip install beautifulsoup4
pip install chardet
pip install html5lib
pip install janome
pip install matplotlib
pip install nltk
pip install numpy
pip install opencv-python
pip install openpyxl
pip install pandas
pip install pillow
pip install plotly
pip install requests
pip install wordcloud
python -c "import nltk; nltk.download('punkt')"
python -c "import nltk; nltk.download('averaged_perceptron_tagger')"
```

上記は packages_pip.sh として、サンプルスクリプトに同梱してあります。実行するには、次のようにシェル sh あるいはコマンドプロンプト cmd を指定します。

```
sh packages_pip.sh                                  # Unix (WSL)
type packages_pip.sh | cmd                          # Windows
```

conda など他のパッケージマネージャを用いるなら、各パッケージのページを参照してください。

外部パッケージのホームページ（あるいはドキュメントページ）は付録 B にまとめて示しました。

目次

はじめに iii

第 1 章　Web スクレイピングとは 1

1.1　Web スクレイピングとは 1
1.2　Web スクレイピングの手順 2
1.3　Web スクレイピングの注意 3
1.4　Web スクレイピングの問題点 4

第 2 章　登場人物のワードクラウドを生成する 5

2.1　目的 5
■ワードクラウド……5　■ターゲット……6

2.2　方法 8
■手順……8　■ターゲットのテキストについて……9　■Requests……10
■NLTK……11　■WordCloud……12　■セットアップ……12

2.3　スクリプト 14
■スクリプト……14　■実行例……15

2.4　スクリプトの説明 16
■概要……16　■get_page……17　■sanitize……18
■不要な文字の削除……20　■文字の置き換え……21　■extract_nouns……22
■pos_tag の癖……24　■generate_wc……26　■WordCloud 画像生成……28

第 3 章　ストーリーラインを描く 31

3.1　目的 31
■ストーリーライン図……31　■ターゲット……32

3.2 方法 33
■手順……33 ■ターゲットのテキストについて……35
■Matplotlib……35 ■セットアップ……37
3.3 スクリプト 38
■スクリプト……38 ■実行例……40
3.4 スクリプトの説明 41
■概要……41 ■CHARACTERS……41 ■sanitize……42
■get_word_positions……43 ■Matplotlib……44 ■generate_plot……47

第 4 章 HTML ページからワードクラウドを生成する 51

4.1 目的 51
■ワードクラウド……51 ■ターゲット……52
4.2 方法 53
■手順……53 ■ターゲットのテキストについて……54 ■Beautiful Soup……56
■Janome……57 ■Mecab IPADIC……58 ■NumPy……60 ■Pillow……60
■セットアップ……61
4.3 スクリプト 61
■スクリプト……61 ■実行例……63
4.4 スクリプトの説明 64
■概要……64 ■get_page の文字化け対策……65 ■extract_text……66
■extract_nouns……68 ■Token オブジェクト……69 ■名詞抽出……69
■整理……71 ■フォントの選択……71 ■背景色と輪郭色……73
■文字色……73 ■型抜き……74

第 5 章 Zip テキストの小説からワードクラウドを生成する.....75

5.1 目的 75
■ワードクラウド……75 ■ターゲット……76
5.2 方法 77
■手順……77 ■ターゲットのテキストについて……78 ■zipfile……79
■Chardet……79 ■Janome Analyzer モジュール……80 ■セットアップ……81
5.3 スクリプト 82
■スクリプト……82 ■実行例……84

5.4 スクリプトの説明 84
■概要……84 ■get_page……85 ■parse_zipped……86
■Chardet による文字エンコーディング推定……87 ■extract_noun……88
■charfilter……89 ■tokenizer……91 ■tokenfilter……91
■CompoundNounFilter……91 ■POSKeepFilter……94
■TokenCountFilter……95 ■Analyzer の実行……96 ■generate_wc……97

第 6 章 HTML の表をグラフにする 99

6.1 目的 99
■グラフ……99 ■ターゲット……100

6.2 方法 101
■手順……101 ■ターゲットの表について……102 ■Pandas……103
■html5lib……104 ■OpenPyXL……105 ■セットアップ……105

6.3 スクリプト 106
■スクリプト……106 ■実行例……107

6.4 スクリプトの説明 108
■概要……108 ■jdate_to_datetime……108 ■extract_tables……110
■DataFrame……111 ■rowspan……113 ■表の結合……114
■注の削除……115 ■日付の変換……116
■マグニチュード値を小数点数に変換……118 ■統計値の計算……118
■generate_plot……119 ■CSV/Excel 形式での保存……122

第 7 章 HTML ページから画像だけを抜き出す 125

7.1 目的 125
■画像オブジェクトの保存……125 ■ターゲット……126

7.2 方法 127
■手順……127 ■ターゲットの画像リンクについて……128
■ターゲットの画像フォーマットについて……129 ■Pickle……130
■Pillow/PIL……131 ■セットアップ……132

7.3 スクリプト 133
■スクリプト……133 ■実行例……134

7.4 スクリプトの説明 136
■概要……136 ■get_page……136 ■多リソースアクセス時の問題……137
■extract_img_links……137 ■絶対 URL の取得……139 ■all_images……140
■画像オブジェクトの属性とメソッド……142 ■画像変換……143
■pickle_save……143 ■show_image……144

第 8 章 HTML ページの画像からアニメーションを生成する 145

8.1 目的 145
■アニメーション画像……145 ■ターゲット……146
8.2 方法 146
■手順……146 ■アニメーション画像フォーマットについて……146
■Pillow の画像保存パラメータ……147 ■セットアップ……148
8.3 スクリプト 149
■スクリプト……149 ■実行例……150
8.4 スクリプトの説明 150
■概要……150 ■image_animation……151 ■キャンバスの生成……152
■貼り付け……153 ■アニメーション保存……153

第 9 章 HTML ページの画像からサムネールを生成する.....155

9.1 目的 155
■サムネール画像……155 ■ターゲット……156
9.2 方法 156
■手順……156 ■台紙画像の構成……157 ■セットアップ……157
9.3 スクリプト 158
■スクリプト……158 ■実行例……159
9.4 スクリプトの説明 159
■概要……159 ■image_thumbnail……160

第 10 章 HTML ページの画像から顔を抽出する.....161

10.1 目的 161
■顔サムネール……161 ■ターゲット……162

10.2 方法 162
■手順 …… 162　■OpenCV …… 163　■画像変換 …… 164
■顔の検出 …… 164　■モデルデータ …… 165
■本書収録のモデルデータ …… 166　■セットアップ …… 167
10.3 スクリプト 167
■スクリプト …… 167　■実行例 …… 169
10.4 スクリプトの説明 170
■概要 …… 170　■detect_faces …… 171　■CascadeClassifier …… 172
■crop_faces …… 174　■get_faces …… 174　■メイン …… 175

第 11 章　REST で取得した地理座標から地図を作成する 177

11.1 目的 177
■インタラクティブマップ …… 177　■ターゲット …… 178
11.2 方法 180
■手順 …… 180　■REST API について …… 181
■JSON テキストについて …… 181　■ターゲットのデータ構造 …… 182
■ターゲットのデータの精度 …… 184　■Plotly Express …… 185
■セットアップ …… 186
11.3 スクリプト 187
■スクリプト …… 187　■実行例 …… 188
11.4 スクリプトの説明 189
■概要 …… 189　■get_page …… 190　■extract_locations …… 191
■generate_map …… 193

第 12 章　CSV の地理座標から地図を作成する 197

12.1 目的 197
■インタラクティブマップ …… 197　■ターゲット …… 199
12.2 方法 202
■手順 …… 202　■抽出する列 …… 202　■文字エンコーディングについて …… 203
■不正な文字について …… 204　■半角カナについて …… 205
■列名について …… 206　■余分な行 …… 207　■余分な列 …… 207
■セットアップ …… 208

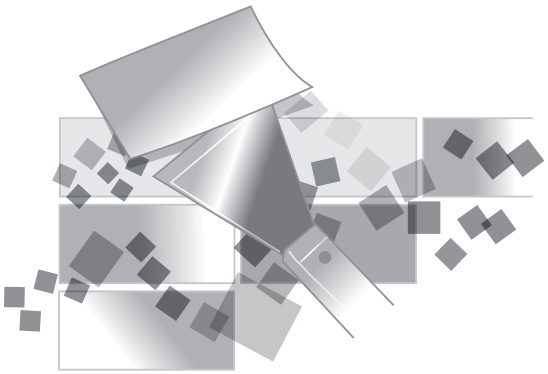
12.3 スクリプト 208
■スクリプト……208　■実行例……210
12.4 スクリプトの説明 211
■概要……211　■get_csv……211
■数値データが正しく解釈されたか確認……212　■列名のスペースの除去……213
■複数の列の値を集約……214　■列の取り出し……215
■NaN の削除……216　■generate_map……217

付 録 219

付録 A　やや高度な話題 219
付録 B　参考文献 253
付録 C　スクリプトリスト 257

索 引 259

1 Web スクレイピングとは



1.1 Web スクレイピングとは

Web スクレイピング、あるいは単にスクレイピングとは、ネットからさまざまなデータをダウンロードし、取捨選択の上で利用することを指します。Scraping の「表面をひっかく」という意味からわかるように、サイトのデータを余すことなく使うのではなく、必要な上澄みだけを利用します。

たとえば、ある商品に興味があるのなら、いくつかのオンラインショッピングサイトから価格だけを抜き出し、それらを揃えて比較します。必要なら、たとえばドル表記の価格を円に揃えるなど、データ変換もします。

選択と変換を経たデータは、わかりやすいように提示します。数値ならグラフにしたり、位置座標なら地図にプロットしたり、画像なら並べて一覧したりします。つまり、まとめ作業です。

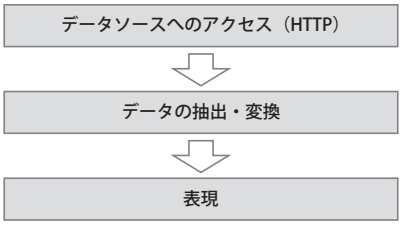
要するに、Web サーフィンで素材を集め、まとめる作業です。普段と異なるのは、人手でかちゃかちゃとコピーしたり集計したりするのではなく、プログラムを使って自動的に処理するところ

です。プログラムを書くのは確かに面倒ですが、データ数が多いと人手がいくらあっても足りません。また、一度書けば、ちょっとした変更だけで似たようなサイトの似たような処理で使いまわせるので、結果的には手間も時間も省けます。

Web スクレイピングは厳密な用語ではないので、人によって定義はまちまちです。「Web」という語が入っている以外は情報の収集と処理にしかすぎないと見ることもできるので、広く取ればコンピュータによる情報処理技術全般と考えることもできます。あるいは技術を絞って、HTTP アクセスと HTML 解析とすることもできます。文章をまとめるのは言語処理で、画像を理解するのは画像処理なので、そうした技術の適用例でもあります。データのグラフィカルな表現はビジュアライゼーションという分野にまとめられるものなので、それを含むか含まないかは意見が分かれます。本書ではやや広い考え方を採用して、ネットからデータを取ってきて、目的に応じて選択と変換を施し、わかりやすいように表現するまでの一連の流れを Web スクレイピングとしています。

1.2 Web スクレイピングの手順

Web スクレイピングは、次に示すステップを通じて行います。



最初はデータソースへのアクセスです。昔と違って、ネットアクセスの方法はほぼ HTTP だけです。REST API もモバイルアプリも、通信プロトコル自体はまず HTTP です。つまり、このステップは Web ブラウザとやることに変わりはありません。違いは、手でクリックするのではなく、データ交換をプログラムで書くところだけです。本書では、主として Requests パッケージを使って HTTP アクセスを実行します。

データはいろいろな形式で表現されているので、それらに応じて解析しなければなりません。HTML ならタグを取り除いたテキスト文を取り出します。データが Zip 形式なら、展開してファイルを取り出します。幸いなことに、Python にはデータ形式（メディアタイプ）に応じた各種のツールが用意されているので、それを使うだけです。本書で利用する外部パッケージを次に示します（括弧内はパッケージ名です）。

データ形式	ツール
HTML	Beautiful Soup 4（bs4）
Zip	Python 標準ライブラリ（zipfile）
表（HTML <table>、CSV、Excel など）	Pandas（pandas）
画像	Pillow（PIL）

データによってはさらなる解析が必要なものもあります。たとえば、日本語テキストならそこから固有名詞だけを抜き出す、画像なら拡大縮小や顔の抽出などの処理です。これにもいろいろなツールがあります。本書では以下の外部パッケージを使います。

解析対象	ツール
英語テキスト	NLTK（nltk）
日本語テキスト	Janome（janome）
画像	Pillow（PIL）、OpenCV2（cv2）

データの解析が終われば、それをわかりやすいように表現します。グラフにする、画像にまとめるなどです。この最後のステップでは、本書では次のツールを用います。

表現	ツール
テキストの画像化	WordCloud（wordcloud）
グラフ	Matplotlib（matplotlib）
画像表示	Pillow（PIL）
地図表示	Plotly Express（plotly）

この 3 ステップからなる手続きは、データベース系の人ならご存じの ETL（抽出、変換、再収容）とほぼ同じです。最後の Load の部分が、データベースに戻すのではなく表現になるところが異なるだけです。何なら理解、分解、再構築と言い換えても構いません。つまり、Web スクレイピングというと新しいテクニックに聞こえますが、昔からある情報処理の手順とたいして変わりはしません。

1.3 Web スクレイピングの注意

Web スクレイピングを禁止しているサイトもあります。明示的に禁止していないサイトでも、短い時間間隔で連続してアクセスすることを禁じたり、そのような挙動が観測されたらアクセスを

ブロックするところもあります。アクセス時には注意してください。

スクリプトの作成時には、テストやデバッグでターゲットに頻繁にアクセスしなければならないこともあります。そうしたときは、Python 標準ライブラリの Pickle を使ってダウンロードしたデータオブジェクトをファイルに落としてそこから利用する、あるいは HTML や CSV などをそのままファイルに保存するなどして、アクセスを最小限にとどめます。Pickle の用法は第 7 章で説明します。

Web サイトのコンテンツは著作権で保護されているものもあります。個人で利用する分にはよいかもしれませんが、成果物を公開するときは注意してください。

1.4 Web スクレイピングの問題点

Web スクレイピングスクリプトは、ターゲットとなるデータソースの構造やフォーマットにもとづいて作成されます。中身を解析するのは、プログラマー本人です。そして、解析が不十分、あるいは例外に対処できていなければ、エラーが発生します。

これはなかなか大変です。データ構造が必ずしも明示されているわけではないので、テストでエラーが発生するたびに、1 つずつつぶしていかなければなりません。Web スクレイピングというと、自動的にほしいものをほしいところから取ってきて整形してくれる便利な方法というイメージがなきにしもあらずですが、そこにたどり着くにはそれなりの労力が必要です。しかも、できあがったと思ったら、データ構造が変わって仕切り直ししなければならないこともあります。

Web スクレイピングスクリプトは、必殺の万能技ではない点、覚えておいてください。

2 登場人物のワードクラウドを生成する

TXT 英語

データソース	プロジェクト・グーテンベルグ
データタイプ	英文テキスト（text/plain; charset=utf-8）
解析方法	固有名詞抽出
表現方法	ワードクラウド
使用ライブラリ	NLTK、Requests、WordCloud

2.1 目的

■ ワードクラウド

本章では、英語で書かれた小説（テキスト文書）から登場人物や場所などの固有名詞を抜き出し、それらを張り付けた画像を生成します。固有名詞は、出現頻度が高いほど大きく、低いほど小さく描き込みます。

次に、ホメロスの『イリアス』から生成した画像を示します。

テキストに含まれている重要な単語をこのように視覚的に表現する技法を、ワードクラウドあるいはタグクラウドと言います。文書内の重要なトピック、物語ならメインキャラが直感的に把握できるという特徴があります。

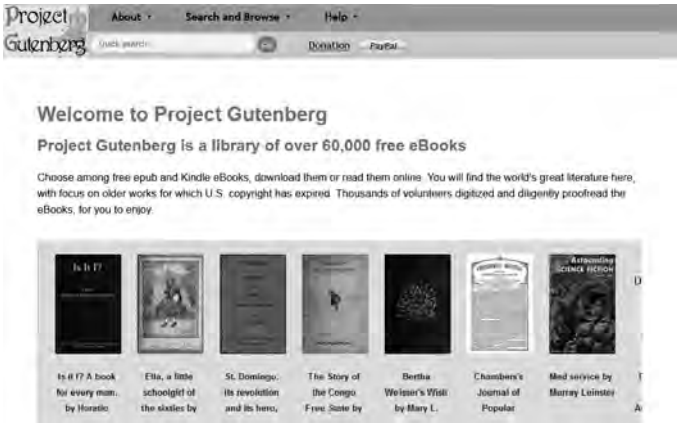
上の例では、最頻の固有名詞は「Achaeans」（アカイア人つまりギリシア人）と「Trojans」（トロイア人）です。ここから、戦争当事者である２つの陣営がわかります。次に多いのは「Jove」（ゼウス）、「Hector」（トロイアのヘクトル王子）、「Achilles」（ギリシア勢のアキレウス）で、戦の勧進元である主神と、それに踊らされて戦う２人のメインキャラです。トロイア戦争ではオデュッセウス（Ulysses）が木馬の姦計で有名ですが、小さく見つからないことから、この話では木馬が出なさそうなこともわかります。

本章で示す方法は、小説以外でも、ある程度のボリュームのあるテキスト文書に使えます（第４章では出版目録をターゲットにします）。ただし、分量がない、あるいは出現頻度に偏りがないテキストでは、さほど効果が得られません。また、頻度が高い単語が重要であるという仮定が成立しない文書では、ミスリーディングにもなります。

ターゲット

例題に用いる『イリアス』は、次に URL を示すプロジェクト・グーテンベルグから入手します。

```
https://www.gutenberg.org/
```



プロジェクト・グーテンベルグはインターネットの黎明期に誕生した、おそらくは最古参の電子書籍サービスです。現在、英語を中心に、著作権の切れた書籍が７万冊以上収容されています。

今回取り上げるホメロスの『イリアス』のテキストには、次に示すように６つの版があります（すべての電子書籍に必ずしも複数の版があるわけではありません。ポピュラーな古典である『イ