半自動着色読書

Semi Automatic Coloring Reading

半自動読書

T.Ishikawa / @polymonyrks　著

2021-07-10 版　　関数型玩具製作所 発行

前書き

本書概要

本書では半自動（セミオート）で読者自身で着色をしながら読書する技術をご紹介します。白黒の文章に「上手に」色を付与すると楽に読書できます。

筆者はここ数年、着色読書の可能性を模索してきましたが、これは間違いないという手法を最近発見しました。前拙著（「技術書を自作 PDF リーダーで読む、Haskell でつくる不思議な読書体験」）を購入くださった方向けですが、そこでご紹介したやり方から大きく前進しています。

また、今回はより手軽に半自動着色読書を体験できるよう、ブラウザ拡張 (Chrome 拡張) を用意しました。Chrome 拡張で確認しながら本書を読み進めて欲しいです。

本技術が洗練されれば、これまでの読書術（速読・精読）が対象としていない類の文書の攻略が楽になると確信してます。とりわけ有効なのは、難文書（専門書等のかっちりした文書、読者が門外漢とする文書、公文書等の形式的な文書）です。本技術の公開がその一助となれば幸いです。

本書はベータ版である件

ご紹介しているソフト（Chrome 拡張）は開発中のものです。技術書典 11 開催期間が終わった際に一旦停止します。

ネタばらしすると筆者の力不足でサーバー周りが貧弱です（多くの人がアクセスした際の挙動が全く不明です）。無料頒布にしたのはその辺りの事情もあります。Chrome ストア対応もできませんでした。サーバーが止まって動かない等の不具合があった際は、上記事情をご考慮くださると幸いです。

また本書自体も準備号です。別の機会（次回技術書典、技術書同人誌博覧会等）で詳細な背景、技術の要点を含めた完全版を頒布予定です。

免責事項

本書に記載された内容は、情報の提供のみを目的としています。したがって、本書を用いた開発、製作、運用は、必ずご自身の責任と判断によって行ってください。これらの情報による開発、製作、運用の結果について、著者はいかなる責任も負いません。

変更履歴

目次

第 1 章

背景

本書では半自動読書という技術をご紹介します。半自動（セミオート）の意味あいですが、半分は機械が担当し、他の半分はヒト（読者）が担当するという意味です。以下の分担になってきます。

- ヒト:色を付けたい要素を選択[*1]
- 機械:ヒトが選択した要素をもとに自動で「いい感じに」着色[*2]

全自動ではなく半自動なのが大事なのですが、それはもう少し後でご説明します。

本章では、文章の着色に成功している例（人工言語）を見たあとで、普通の文章（自然言語）を着色する際の課題、半自動読書によるその課題解決を見ていきます。最後に半自動読書の一つの要となる「効果的な着色単位」について触れます。

1.1 着色の成功例 - シンタクスハイライト

着色読書の成功例としてプログラミングのシンタクスハイライト (Syntax Highlighting) が挙げられます。白黒のメモ帳でプログラミングする人はいない？ はずです。プログラムをしない人もどちらが読みやすいか、見比べてみてください。

[*1] 正確にはこれです。自身の了解する着色戦略に基づき注目したい要素を指定（ただし読書への集中を欠かない最低限で）

[*2] 正確にはこれです。「効果的な着色単位の決定（構文解析）」、「重要部の提案（サジェッション）」、「ヒトが指定した箇所を自動で（網羅的に高速で）着色」など

```
map' :: (a -> b) -> [a] -> [b]
map' f [] = []
map' f (x : xs) = f x : (map f xs)

foldl' :: (a -> b -> a) -> a -> [b] -> a
foldl' f x0 [] = x0
foldl' f x0 (x : xs) = foldl' f (f x0 x) xs

filter' :: (a -> Bool) -> [a] -> [a]
filter' f [] = []
filter' f (x : xs)
  | f x = x : next
  | otherwise = next
    where
      next = (filter' f xs)
```

▲図 1.1　白黒のコード

```
map' :: (a -> b) -> [a] -> [b]
map' f [] = []
map' f (x : xs) = f x : (map f xs)

foldl' :: (a -> b -> a) -> a -> [b] -> a
foldl' f x0 [] = x0
foldl' f x0 (x : xs) = foldl' f (f x0 x) xs

filter' :: (a -> Bool) -> [a] -> [a]
filter' f [] = []
filter' f (x : xs)
  | f x = x : next
  | otherwise = next
    where
      next = (filter' f xs)
```

▲図 1.2　シンタクスハイライトされたコード

シンタクスハイライトは文書への着色が最も成功している例とみてます。

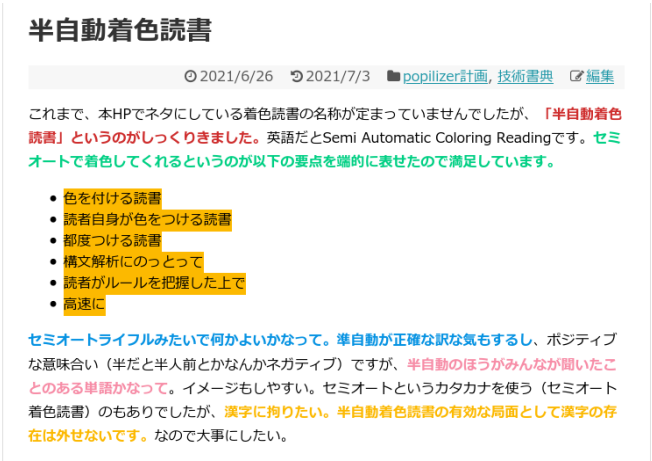
プログラミング言語は人工言語です。文法が厳格でそれに基づいた着色も例外なく整然と行われます。それ故に着色による視認性向上に成功していますが、自然言語、すなわち

普通の文章ではどうなるでしょうか？ 人工言語と異なり文法がゆるい分、一歩間違えると視認性が低下したり読書への集中力が切れたりしがちです。以下、自然言語に対する着色の抱える課題を一つずつ確認していきます。

1.2 自然言語に対する着色の抱える課題

1.2.1 意図の読めない着色

筆者が思うがままに着色していて読者がその着色の意図を汲めない場合、白黒のままのほうがいいかなって思ったりします。



▲図 1.3 カラフルに彩られたブログ

読者の助けとなるように着色されている場合も、それにありがたみを感じる読者がいる一方で、お節介と感じる読者もいるはずです。

1.2.2 他人による着色

友達から手渡された教科書にカラフルなマーカーがびっしり引かれてた時を思い出してみてください。普段マーカーを引かない人だけでなく、普段マーカーを引く人も良い印象を持たないのではないでしょうか。これは、その着色が「他人によってなされたという事実」への動物的嫌悪感です[*3]。犬の散歩中のマーキングと似ています。他の犬にされてし

[*3] ここには先述の「意図が読めない着色」問題もあります

まったマーキングを上書きしたくなる本能です。読書の話に戻すと、何もないところから読者自身で着色していくのが良さそうです。[*4]。

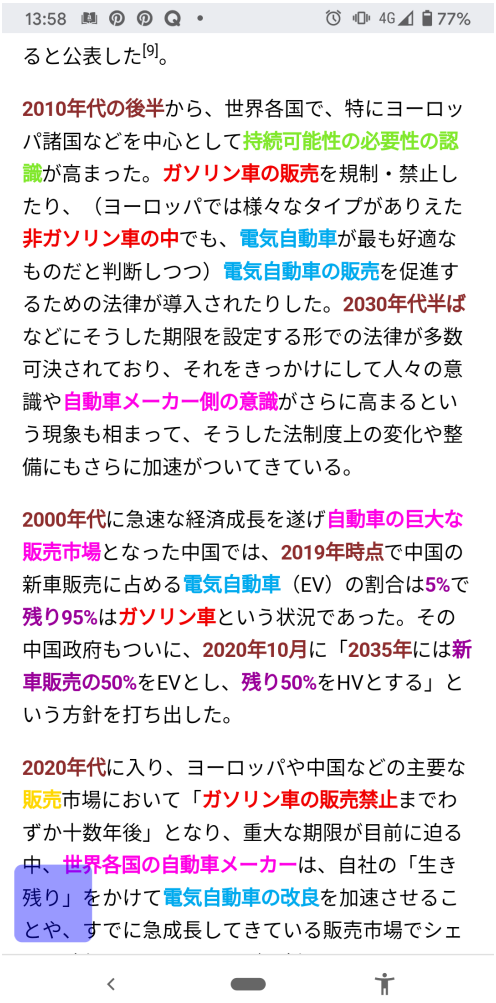
1.2.3 既に完成された着色

マーカーの例が続きますが、自分で引いたけど広範囲にびっしりマーキングした場合を想定してみてください。さらに少し時間が経った、例えば試験のためにマーカーを引いて試験後数日後たったとします。見返すとどうでしょう、人によっては既に完成されたカラフルな教科書に圧倒されたりしそうです。

マーカーを引くときは、真っ白な状態からスタートして段階を踏んで引いていきます。その手順、マーカーを引くという行為が大事であって、完成形が大事ではない人の場合は、時間をおいたあとのびっしりマーキングは面白くないです。

マーカーとは少しずれますが、以下の図を見てください。

[*4] それではシンタクスハイライトはどうかというと、「ほぼ一意に定まる」のが効いてきそうです。細かいことを言うとシンタクスハイライトでも冗長 (Verbose) なものを好まないケースもあります



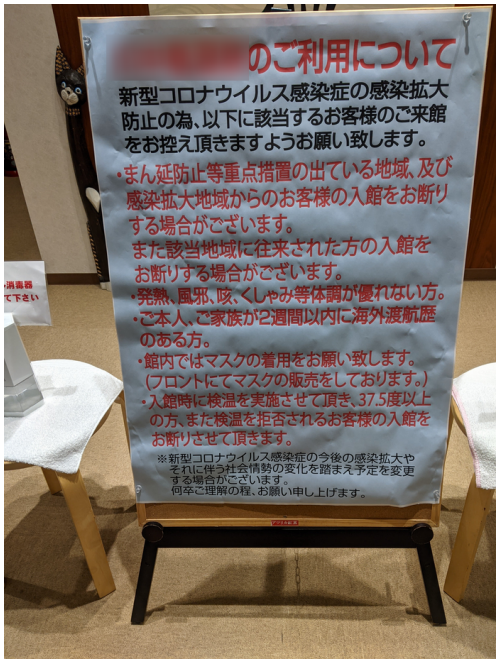
▲図 1.4　チュートリアルの最終型

広範囲に色が展開しています。これは後ほどのチュートリアルの最終型です。かなり目がチカチカします。しかしチュートリアルでは段階を踏んで最後にこの形になります。チュートリアルをやってから少し時間が経ったあとに上の画像を見てみると面白かもしれません。

1.2.4　バルク的な着色単位

全て赤字の文章を見たりしませんか？[*5]。全体が大事なのは分かりますが、文章の詳細を読む観点では視認性は向上していないです。読者が赤字に慣れていない場合には、視認性が低下しかねないです。

全体を赤にするのではなく、少し手間ですが小分けすると読みやすくなります。どの単位で小分けするとよいかは、この章の後半で詳しく説明します。



▲図 1.5　赤字の立て看板

1.3　半自動読書による課題解決

自然言語を着色する際に押さえるべき点をまとめます[*6]。上記問題点を裏返しました。

[*5] この立て看板のある場所は筆者のお気に入りのスポットです。よく行くので看板がいつも気になってしまいます。

[*6] 他の観点では論文等の専門性の高い文書においては白黒の文章であることが多いです。これは着色することで却って視認性が低下したり集中力を切らしたりするデメリットを考慮してそうです。半自動読書はこ

- 着色の意図を読者が了解する
- 着色は読者自身が行う
- 着色は都度少しずつ完成させる
- 着色はいい感じの単位で行う

半自動読書はこれらを可能としています。チュートリアルの章で実際にソフトを動かしながら確認してほしいです。

1.4 いい感じの着色単位

チュートリアルの前に、上で出てきた「いい感じの着色単位」についてご説明します。自然言語には、人がものごとを認識する上で自然な単位が存在すると筆者はみています。

1.4.1 着色例

下の 2 つの強調例を見てください。技術書典 9 の利用規約[*7]の一部を抜粋しました[*8]。

技術書典に参加するサークルは**技術書の**頒布を目的として参加ください。

- 頒布形態のメインは電子書籍（デジタルコンテンツ）です。紙の書籍（いわゆる同人誌）の**サポートもしています。**
- オンラインマーケットでは**紙の書籍を用意するときは必ず「電子書籍＋紙」として同一内容の電子書籍を用意頂いています。**
- 頒布物は**全年齢向けが対象です。成人向け書籍は対象外です**
    - 対象外となる書籍は「禁止事項」の項目を確認してください
- オンラインマーケットでは書籍審査があります
    - 参加費を無料にしたり、販売手数料を無料にしたりと間口を広げる工夫をしたので書籍の審査によっていわゆるマルチ商材などの販売を防ぎます
    - 詳細は規約から技術書典オンラインマーケット販売サービス利用規約をご確認ください
    - 技術書かどうかは不明なんだけど…というケースは必ずお問い合わせください
    - オリジナル、商業ベースの技術書も頒布可能です
    - 書籍は本棚に登録されて管理されます。本棚の都合、チラシや書籍の一部（サンプル）などの**配布は行なえません**（サンプルはサムネイルに追加してください）

▲図 1.6　技術書典 9 - 利用規約

の領域にも切り込んでいきます。

[*7] 技術書典 9 - 利用規約 (https://techbookfest.org/event/tbf09#requirements)

[*8] この着色単位の話は前著（技術書典 9 で刊行）からの抜粋です

技術書典に参加するサークルは**技術書の頒布**を目的として参加ください。

- 頒布形態のメインは電子書籍（デジタルコンテンツ）です。**紙の書籍（いわゆる同人誌）のサポート**もしています。
- オンラインマーケットでは**紙の書籍**を用意するときは必ず**「電子書籍＋紙」**として**同一内容の電子書籍**を用意頂いています。
- 頒布物は**全年齢向けが対象**です。**成人向け書籍は対象外**です
  - 対象外となる書籍は「禁止事項」の項目を確認してください
- オンラインマーケットでは書籍審査があります
  - 参加費を無料にしたり、販売手数料を無料にしたりと間口を広げる工夫をしたので書籍の審査によっていわゆるマルチ商材などの販売を防ぎます
  - 詳細は規約から技術書典オンラインマーケット販売サービス利用規約をご確認ください
  - 技術書かどうかは不明なんだけど…というケースは必ずお問い合わせください
  - オリジナル、商業ベースの技術書も頒布可能です
  - 書籍は本棚に登録されて管理されます。本棚の都合、**チラシや書籍の一部（サンプル）などの配布**は**行なえません**（サンプルはサムネイルに追加してください）

▲図 1.7　技術書典 9 - 利用規約（別の強調）

特に気をつけるべき点をどちらも強調していますが、下のほうが自然な区切りで読みやすい気がしませんか？

次は英語の例です。Wikipedia の S 式の説明[*9]です。英語は単語がスペースで区切られていますが、もう少し広い範囲で自然な区切り、単位が存在します。

[*9] S 式 (wikipedia)(https://en.wikipedia.org/wiki/S-expression)

In computer programming, S-expressions (or symbolic expressions, abbreviated as sexprs) are a notation for nested list (tree-structured) data, invented for and popularized by the programming language Lisp, which uses them for source code as well as data. In the usual parenthesized syntax of Lisp, an S-expression is classically defined as

an atom, or

an expression of the form (x . y) where x and y are S-expressions.

The second, recursive part of the definition represents an ordered pair, which means that S-expressions can represent any binary tree, though S-expressions which contain cycles cannot conversely be represented as binary trees.

▲図 1.8　S 式 (wikipedia)

In computer programming, S-expressions (or symbolic expressions, abbreviated as sexprs) are a notation for nested list (tree-structured) data, invented for and popularized by the programming language Lisp, which uses them for source code as well as data. In the usual parenthesized syntax of Lisp, an S-expression is classically defined as

an atom, or

an expression of the form (x . y) where x and y are S-expressions.

The second, recursive part of the definition represents an ordered pair, which means that S-expressions can represent any binary tree, though S-expressions which contain cycles cannot conversely be represented as binary trees.

▲図 1.9　S 式（wikipedia, 別の強調）

いずれも S 式の特徴を表す箇所を強調していますが、下の例のほうが読みやすくないでしょうか？

an ordered pair, S-expressions, any binary tree あたりは、もしそれ専用の wiki ページのリンクがあれば、上記下図のような区切りで着色されるはずです。上記第二の例が英語の場合の自然な単位です。

1.4.2　自然な単位の正体

名詞句という単位

それでは自然な単位とは何か。個々で言う自然な単位は、名詞句という単位と一致しています。では名詞句とは何か。名詞句は以下のとおりに、該当箇所を「それ」で置き換えても文法上の不自然さがないものです。

技術書典に参加するサークルは**それ**を目的として参加ください。

- 頒布形態のメインは電子書籍（デジタルコンテンツ）です。**それ**もしています。
- オンラインマーケットでは**それ**を用意するときは必ず**それ**として**それ**を用意頂いています。
- 頒布物は**それ**です。**それ**です
  - 対象外となる書籍は「禁止事項」の項目を確認してください
- オンラインマーケットでは書籍審査があります
  - 参加費を無料にしたり、販売手数料を無料にしたりと間口を広げる工夫をしたので書籍の審査によっていわゆるマルチ商材などの販売を防ぎます
  - 詳細は規約から技術書典オンラインマーケット販売サービス利用規約をご確認ください
  - 技術書かどうかは不明なんだけど…というケースは必ずお問い合わせください
  - オリジナル、商業ベースの技術書も頒布可能です
  - 書籍は本棚に登録されて管理されます。本棚の都合、**それ**は**それ**（サンプルはサムネイルに追加してください）

▲図 1.10　技術書典 9 - 利用規約（「それ」で置き換え）

必須条件

この、「それ」は指示代名詞と呼ばれます。指示代名詞で置き換え可能なのが名詞句です。

利用規約のうまく着色されていない例を再び見てみます。

技術書典に参加するサークルは**技術書の**頒布を目的として参加ください。

- 頒布形態のメインは電子書籍（デジタルコンテンツ）です。紙の書籍（いわゆる同人誌）の**サポートもしています。**
- オンラインマーケットでは**紙の書籍を用意するときは必ず「電子書籍＋紙」として同一内容の電子書籍を用意頂いています。**
- 頒布物は**全年齢向けが対象です。成人向け書籍は対象外です**
    - 対象外となる書籍は「禁止事項」の項目を確認してください
- オンラインマーケットでは書籍審査があります
    - 参加費を無料にしたり、販売手数料を無料にしたりと間口を広げる工夫をしたので書籍の審査によっていわゆるマルチ商材などの販売を防ぎます
    - 詳細は規約から技術書典オンラインマーケット販売サービス利用規約をご確認ください
    - 技術書かどうかは不明なんだけど…というケースは必ずお問い合わせください
    - オリジナル、商業ベースの技術書も頒布可能です
    - 書籍は本棚に登録されて管理されます。本棚の都合、チラシや書籍の一部（サンプル）などの**配布は行なえません**（サンプルはサムネイルに追加してください）

▲図 1.11　技術書典 9 - 利用規約（「それ」で置き換えできない）

置き換え不可な箇所だけ「赤」で残しました。名詞句縛りを入れたのと比べると統一感がない印象です。

技術書典に参加するサークルは**技術書の**頒布を目的として参加ください。

- 頒布形態のメインは電子書籍（デジタルコンテンツ）です。紙の書籍（いわゆる同人誌）の**サポートもしています。**
- オンラインマーケットでは**紙の書籍を用意するときは必ず「電子書籍＋紙」として同一内容の電子書籍を用意頂いています。**
- 頒布物は**全年齢向けが対象です。成人向け書籍は対象外です**
    - 対象外となる書籍は「禁止事項」の項目を確認してください
- オンラインマーケットでは書籍審査があります
    - 参加費を無料にしたり、販売手数料を無料にしたりと間口を広げる工夫をしたので書籍の審査によっていわゆるマルチ商材などの販売を防ぎます
    - 詳細は規約から技術書典オンラインマーケット販売サービス利用規約をご確認ください
    - 技術書かどうかは不明なんだけど…というケースは必ずお問い合わせください
    - オリジナル、商業ベースの技術書も頒布可能です
    - 書籍は本棚に登録されて管理されます。本棚の都合、チラシや書籍の一部（サンプル）などの**配布は行なえません**（サンプルはサムネイルに追加してください）

▲図 1.12　技術書典 9 - 利用規約（やや冗長）

逆に「それ」で置き換え可能な箇所を「赤」で残しています。が、今度は赤で強調している箇所が長すぎて、インパクトが薄まっている感があります。名詞句でも程々の長さである必要がありそうです。

英語でも見ておきましょう。英語の指示代名詞は"it"です。"it"で置き換えました。

In computer programming, S-expressions (or symbolic expressions, abbreviated as sexprs) are a notation for nested list (tree-structured) data, invented for and popularized by the programming language Lisp, which uses them for it as well as it. In the usual parenthesized syntax of Lisp, an S-expression is classically defined as

an atom, or

an expression of the form (x . y) where x and y are S-expressions.

The second, it of the definition represents it, which means that it can represent it, though S-expressions which contain it cannot conversely be represented it.

▲図 1.13　S 式（wikipedia, "it"で置き換え）

文法的に問題ないことがわかります。

In computer programming, S-expressions (or symbolic expressions, abbreviated as sexprs) are a notation for nested list (tree-structured) data, invented for and popularized by the programming language Lisp, which uses them for source code as well as data. In the usual parenthesized syntax of Lisp, an S-expression is classically defined as

an atom, or

an expression of the form (x . y) where x and y are S-expressions.

The second, recursive part of the definition represents an ordered pair, which means that S-expressions can represent any binary tree, though S-expressions which contain cycles cannot conversely be represented as binary trees.

▲図 1.14　S 式（wikipedia, "it"で置き換え不可と冗長）

最初の例は、uses, for source code as well as data の箇所は it で置き換えられません。

さらに、下の強調部はかろうじて it で置き換え可能ですが、長すぎます。

パワーポイント等の資料を作る際に、色を変えて強調することがあるはずです。その際に手間ではありますが上記自然な単位を考慮して強調すると視認性が格段に違ってきます。この目線でお店の注意書きの張り紙を観察するのも面白いです。

1.5 まとめ

半自動着色読書を可能とする Chrome 拡張はこの単位で着色がなされます[*10]。Chrome 拡張は popilizer と名付けました。以下簡単のために本 Chrome 拡張のことを popilizer と呼んでいきます。次章以降でインストール方法と具体的な使い方を見ていきます。

[*10] 半自動読書技術としてここが一番難しいし面白いです。なになにの観点で置き換え可能、なになにという文脈で置き換え可能、というのは哲学的にも数学的にも面白いネタが転がっているとみています。

第 2 章

ソフトのインストール方法

本章では popilizer のインストール方法をご紹介します。

2.1　ソフトを動かすために必要なもの

- Android 端末
- KiwiBrower, Yandex 等の Chromium 系ブラウザのインストール

2.2　Android 端末スペック（ご参考）

ソフトを動かすために必要なスペックですが、ご参考ですが、筆者は以下の環境で開発しています。

- 機種: Pixel 3a　（スマートフォン）
- CPU: Snapdragon 670

- 機種: Matepad pro　（タブレット）
- CPU: Kirin 990

Kirin 990 は Snapdragon 7xx などのミドルレンジ帯と置き換えてください。Snapdragon 8xx レベルなら余裕のはずです。

印象としてはメモリと言うより CPU 性能が効いてそうです。具体的には DOM 要素の構築速度が律速になってきます。

2.3 何をやっているソフトか

ブラウザが HP 内の文字列を取得し、それをサーバーに投げる。サーバーは構文解析をし結果をブラウザに返す。ブラウザはそれを元の位置に戻す。そういうフローです。元の位置に戻す際の DOM 構築に負荷がかかります。更にタップで対応する位置の色が変わるのですが、その際の DOM 更新（レンダー）にも時間がかかります[*1]。

2.4 詳細なインストール方法

インストールの詳細については、下記サイト[*2]を参考にしてください。

[*1] サーバーに関しても仮に大人数がアクセスしたには挙動が不明ではあります。

[*2] https://polymony.net/2021/07/05/popilizerInstallation/

第 3 章

操作方法と戦略

本章では popilizer の使い方を説明します。Wikipedia の「電気自動車」のページ[*1]をネタにします。KiwiBrowser で開いておいてください。

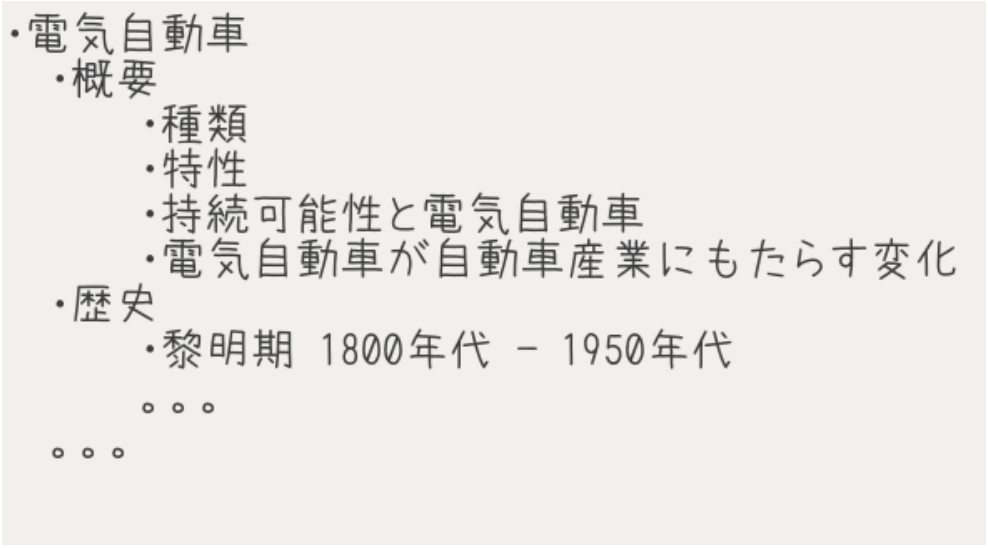
なお、本チュートリアルと違う感じで色を付けてしまった場合には、ページのリロードで最初に戻ってください。

3.1　着色戦略 1（重要語の決定）

着色する際には何を着色するかを指定する必要があります。着色するのは重要な単語です。では何が重要語か。これを特定するのにあまり時間をかけすぎると、読書からの集中が切れます。できればノータイムで指定したいです。

最初の、そして最も重要な着色戦略ですが、重要語はその文章の表題から選んでください。例としてこの「電気自動車」の記事の大項目、小項目を抜き出しました。

[*1] https://ja.m.wikipedia.org/wiki/%E9%9B%BB%E6%B0%97%E8%87%AA%E5%8B%95%E8%BB%8A

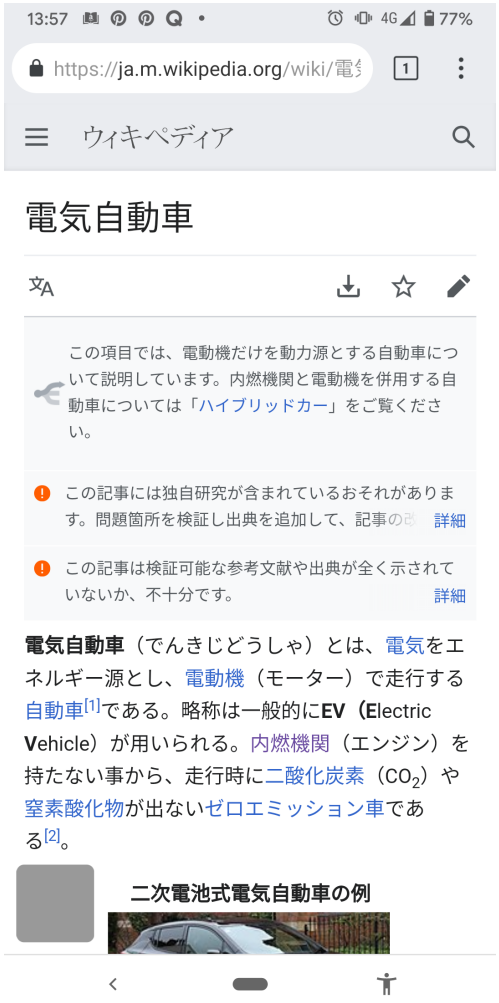


▲図 3.1　大項目、小項目

ページをスクロールしていくと、これらが上から順に出てきます。これらを元に着色する語を選んでいきます。それでは読んでいきましょう。

3.2　開始時の状態

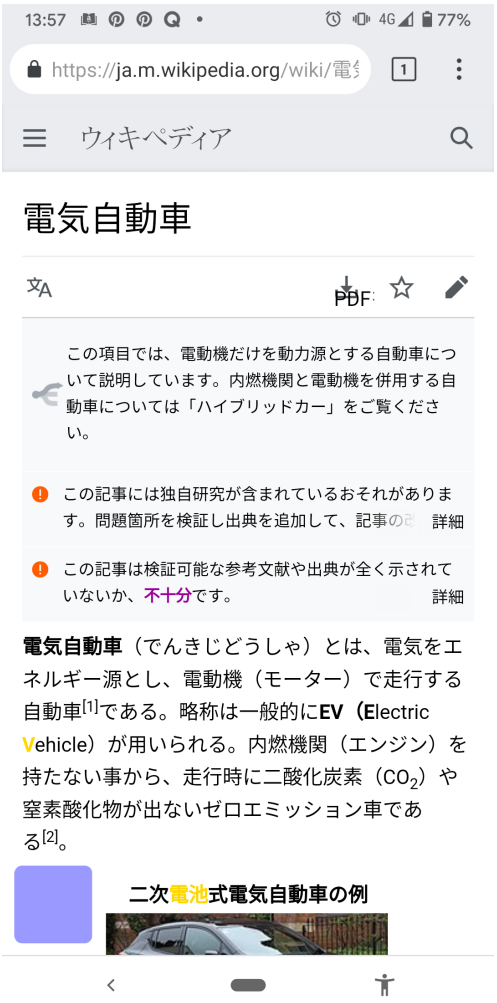
開始時は下図のとおりになっているはずです。



▲図 3.2　開始時

左下に半透明のボタンが見えています。これが popilizer の起動ボタンです。押してみましょう。

3.3 popilizer 起動直後の状態



▲図 3.3　popilizer 起動直後

popilizer が起動します。起動すると半透明のボタンが青色になります。本文中の変化ですが、以下の４つがなされます。

- 文章強調がオフ

- リンククリックがオフ
- 重要語が黄色に（後述）
- 日付系が茶色、物理量系が紫色に（後述）

3.3.1 文章強調のオフ

色を付けていきたいので、初期状態が黒（背景が黒系の場合は白）で統一されます。

3.3.2 リンククリックのオフ

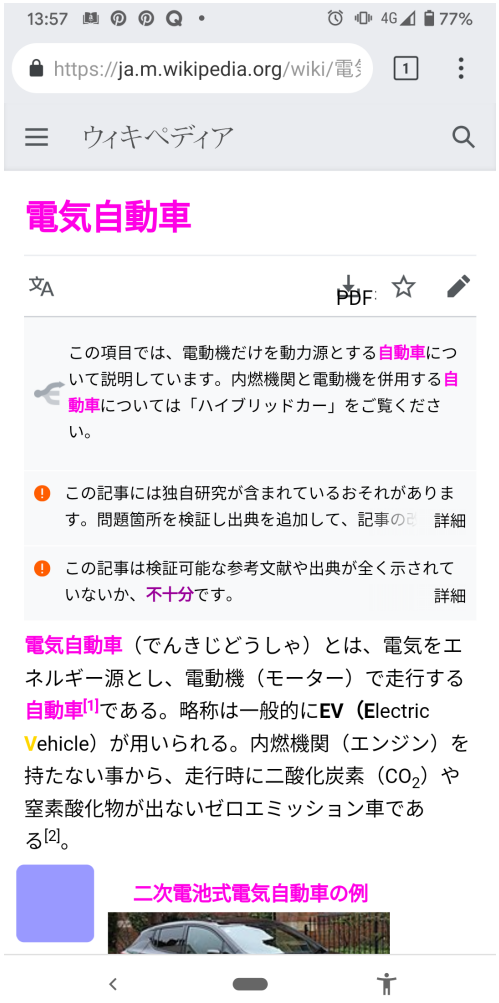
全て白黒になっていてもリンクは存在します。着色は文章の一部をクリック（タップ）することでなされます。クリック時にリンク遷移されるとマズいので、popilizer 起動時にはそれがオフになります。更にリンク部は得てして太字強調と下線が引かれているので、それもオフにしています。モードを切り替えることでリンク遷移できるようにしています。後ほどご紹介します。

3.3.3 その他着色

黄色、茶色、紫色はここでは無視しておいてください。次に進みます。

3.4 最初の着色

これで着色する準備が整いました。表題が「電気自動車」となっています。「電気」ではなく「自動車」の部分をクリックしてみてください。

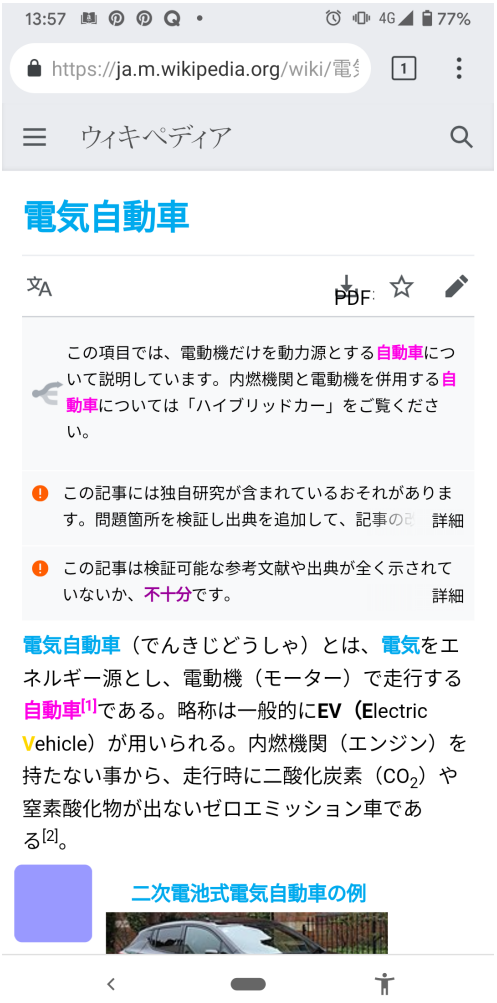


▲図 3.4　最初のクリック（「自動車」）

対応する箇所がピンク色になります。

3.5　着色の仕組み１（上書きされる着色）

次は「電気」をクリックしてみてください。



▲図 3.5　上書きされた着色

電気を含む箇所が水色になっています。「電気自動車」の箇所をご覧ください。最初はピンク色でしたが水色になっています。このように後付の設定が優先的に適用されます[*2]。

[*2] 電気を指定した後で自動車を指定する場合は自動車側の設定が優先されます。

3.6　着色戦略 2（形容詞＋名詞の場合は名詞を先に）

注目したい単語、今回であれば表題の「電気自動車」ですが、このように形容詞（電気）＋名詞（自動車）の構成の場合は、名詞を先にクリック、次に形容詞をクリックするのが有効です。

「自動車」は色々ある、その中でもとりわけ「電気」を使っているものを知りたい、というのがこの Wikipedia の記事です。

自動車的要素（ピンク色）をそうでない箇所（黒字）と区別する、これだけでも有効だったりします。

更にその「自動車（ピンク色）」のなかでも「電気自動車（水色）」を区別したい。着色は上書きしてなされるので「電気」を後にクリックすることになります。これが「形容詞＋名詞ルール」です。

3.7　着色の仕組み 2（色のトグル）

色はピンク、水色、ライム色、オレンジ、赤、青、緑、デフォルト（黒、ボールドなし）の順に変わっていきます。これは色を選ぶ手間を省くためです。着色読書は読書からの集中を切らす側面があります。そういう要素を取り除く必要があります。色を選ぶ自由度は持つべきでないです[*3]。シンタクスハイライトを思い出してください。「どの要素が何色か」、が大事ではなくて、「各要素同士が区別されている」、が大事です。自然言語も同様です。それでは読み進めていきましょう。

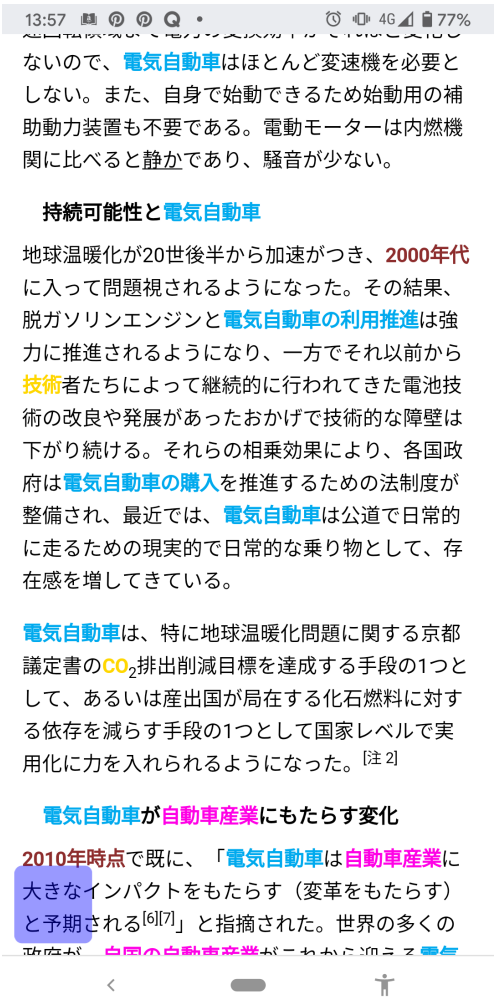
3.8　Tips（色を付けない要素）

「概要」、「種類」、「特性」という小項目と出会いますが、これらは色を付けなくてもいい系です。これは慣れてきたらでいいので Tips としています[*4]。気になる人は下の脚注を読んでみてください。更に進みます。

[*3] 共感覚路線もありえます。少し試したことがあります。ただし読者自身の能力とともにソフト側もよほど賢くならないと無理と分かったので凍結しています。

[*4] 上に述べたとおり、着色はトグルしてなされます。なので有効でない＝そんなに出現してこない単語に色を割くとすぐに使い切ってしまいます。この、「概要」等はいずれも固有名と言うよりは属性の説明的な表題なので、恐らく出てこないと読めます。話を戻すとそれでは色を使い切ったらどうするか。もう一巡するかリセットするかです。後者は最後の方で触れます。

3.9　Tips（色を付けた方がいい要素）

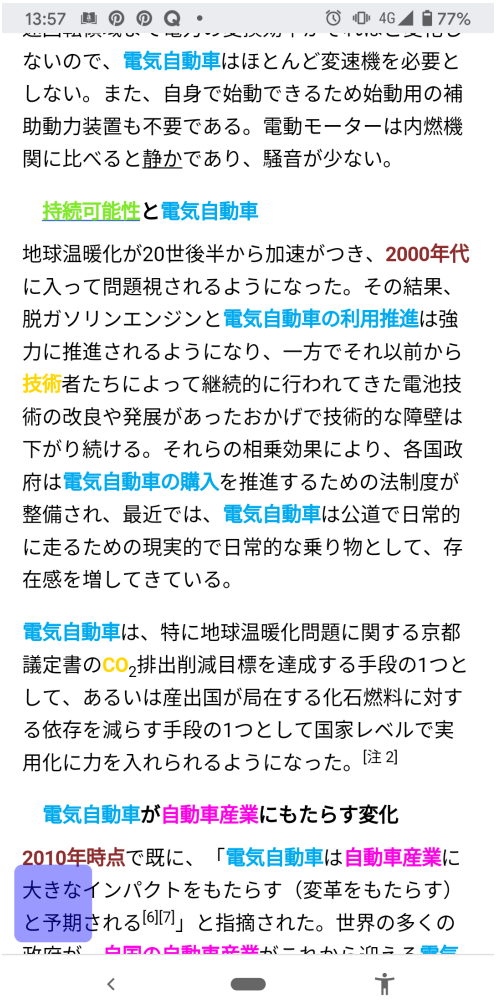


▲図 3.6　色を付けたほうがいい要素

「持続可能性と電気自動車」という項目に出会いました。これは色を付けたほうがいい系です*5。電気自動車側はすでに着色しているので「持続可能性」を着色します。「持続」

*5 理由が気になる人は「色を付けない要素」の脚注を参照ください

と「可能性」の二つがありそうですが[*6]、以下、説明のために「持続」ですすめます。「持続」をクリックしてください。

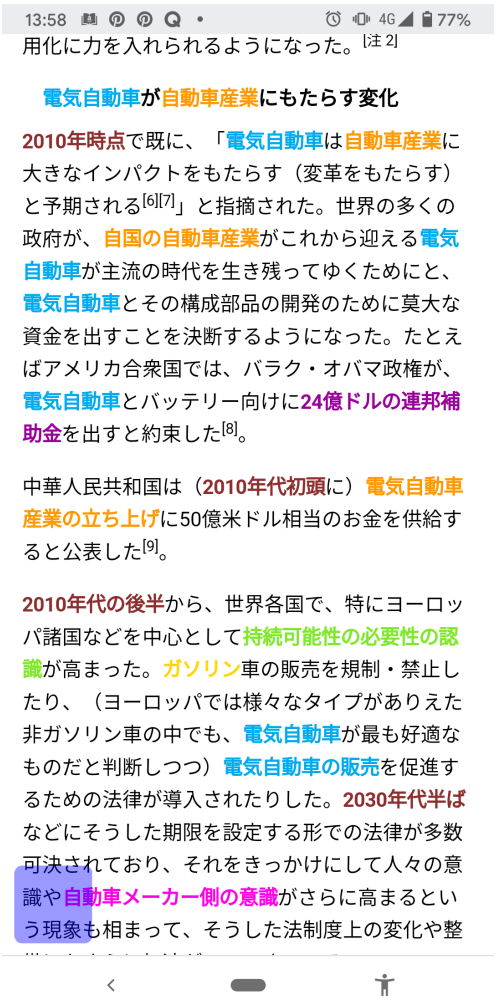


▲図 3.7　結果としてあまり色がつかなかった

はい、残念ながらそんなに色（ライム色）は付きませんでした。こういうこともあります。同じ図で下の方に「電気自動車が自動車産業にもたらす変化」という項目が見えてい

[*6] 何故この二つに分節されるかは、、すいません慣れというのが答えになってきます。チュートリアルが終わったあとで色々クリックして確認してみてください

ます。「産業」に色を付けたくなります*7。なのでクリックします。



▲図 3.8　オレンジになった

はい、オレンジ色になりました。同じ図で茶色と、むらさき色がありますね。

- 茶色（2010 年時点、2010 年代初頭、2010 年代の後半、2030 年代半ば）

*7 「電気」、「自動車」は既に着色しているので、新規に着色するなら「産業」か「変化」になってきます。前者はこのあとで説明が出てきそうな感がある一方で、後者は既に出てきた「概要」などと同様のカテゴリです（色を付けなくてもいい系）。

- むらさき色（24 億ドルの連邦補助金）

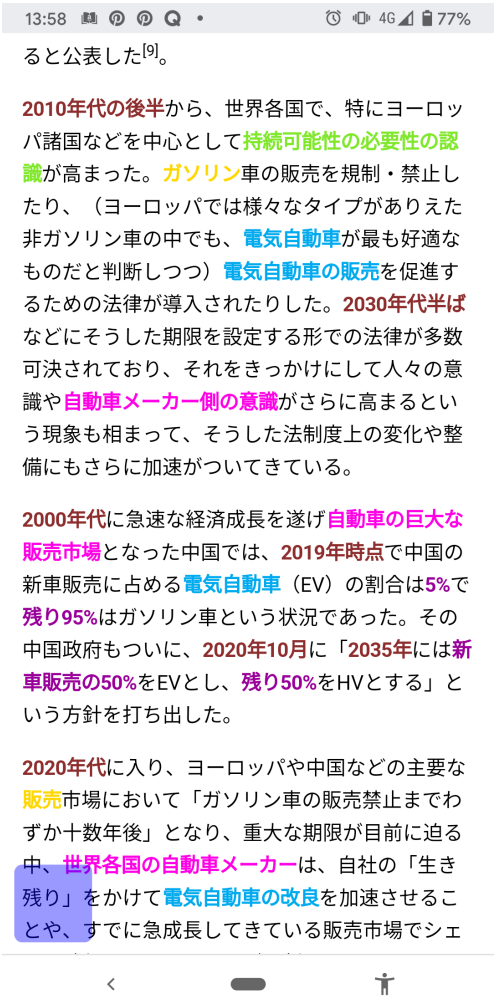
となっています。

3.10　着色の仕組み 3（プリセット語）

自然言語を着色して読む際の難点として、読者が着色の意図を理解していないと×となってましたが、逆に言えば着色の意図を理解できるなら予め着色（プリセット）しておいたほうが、「何を着色するか考える」コストが消せるので○です。そして、時間系や数値系を着色しておくと読みやすくなるのが分かっています。以下の対応関係です。

- 茶色（時間系）
- むらさき色（数値系）

少し読み進めた絵も載せます。



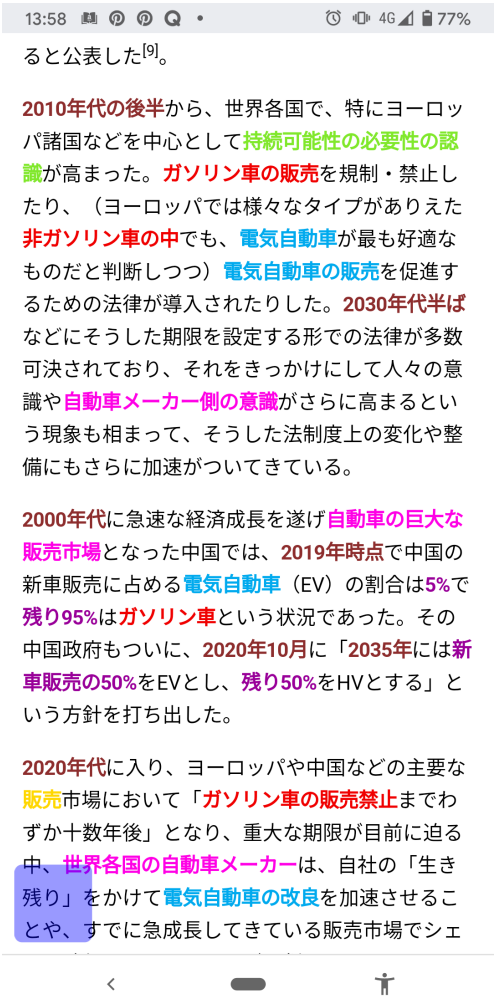
▲図 3.9　プリセットの説明図（時間と数値系）

% とかも紫になっています。popilizer の今の設定だとこれらがプリセットされています。

3.11 Tips（ヒント語の利用)

上のプリセットの説明図を引き続き使いますが、「ガソリン」が黄色になっています。これは「重要語」です。あるブロック中での出現頻度を元に割り出しています[*8]。今回はあまり重要ではないですが一応の説明です。Wikipedia の場合は基本的には着色戦略１で進めてくだい。一応クリックしておきます。

[*8] 機械学習で重要語を割り出すのもやってみたのですが、単純なヒストグラムが結局有効でした。これは色を付けるという観点では、「色のついた面積が広い」というのが大事になるからです。それはヒストグラムと一致してきます（細かく言うととり方は複数種あるのですが）



▲図 3.10　ヒント語の利用

ガソリンらしく赤になりました（笑）[*9]。ちゃんと広範囲が赤になっていますね。

[*9] 共感覚路線だとこういうのが大事になってきますが、それはまた別の機会に。

3.12　着色戦略 3（着色の出来栄え評価の禁止）

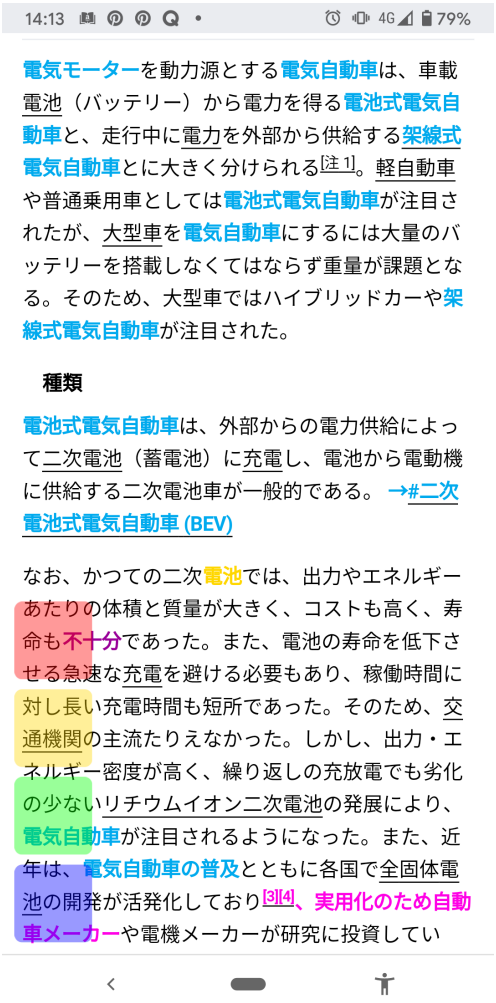
これが最後の戦略です。逆説的かもですが、色が上手くついたか上手くついていないかは無視しましょう。上で言うと、「ライム色（「持続」）があんまり色がつかなかったなぁ、というやつです。着色は読書を楽にするものですが、着色をメタ的な視点で評価し始めると肝心の本文からの集中が切れます[*10]。本チュートリアルはふむふむ、と体験してもらいたいですが、ご自身で色んなサイトで実験する際には、ここで言っていることは空気を吸うようにやれるようトレーニングが必要です。冒頭に述べた自転車に乗るのと同じと思ってください。「ペダルを漕ぐ」、「ブレーキを踏む」というアクションは意識してないはずです。下手に意識するとコケたりします[*11]。

[*10] 実は筆者（開発者）はこの点ではかなり不利です。いちいち出来栄えを確認したり改善点を思いついたりして集中が切れます。この開発を始めてから白黒の紙の読書も以前よりも集中できなくなりました orz...

[*11] プリセット語や名詞句単位のチューニングが肝になります。まだまだ改善が必要です。

3.13　着色の仕組み 4（リンク表示復活と着色リセット）

3.13.1　リンク表示復活



▲図 3.11　リンク表示復活

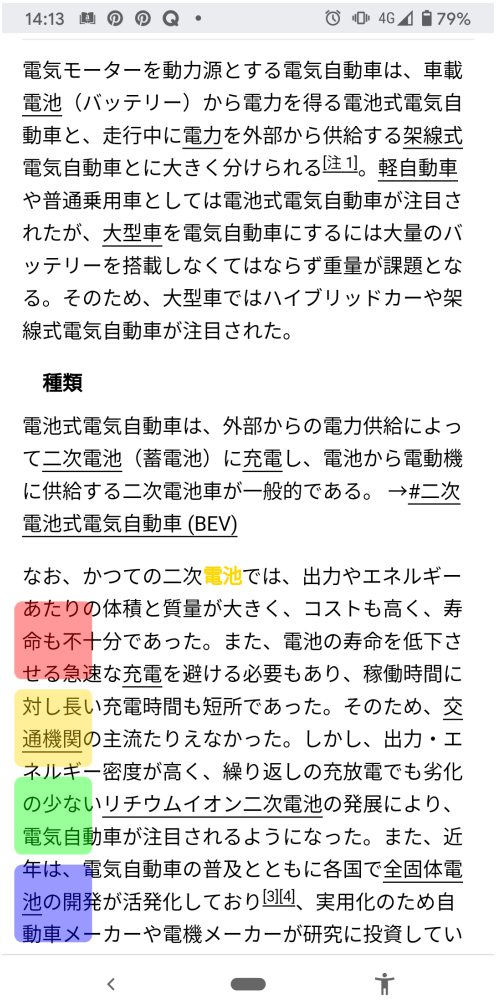
ここからは読書外のことです。popilizer を起動するとリンク位置がわからなくなってしまいます。リンク位置を再出現するには左下の青半透明ボタンを押してください。する

と、リンク箇所に下が引かれます。これをシングルクリックするとリンク遷移できます。

3.13.2 着色リセット

色を使い切った場合や、あまりに多く色が付きすぎた場合はページのリロードが一つの手段ですが、青半透明ボタンの上に出てきた赤ボタンを押すことで色リセットできます[*12]。

[*12] (上の画像はちょっと違ってますが無視してください)



▲図 3.12　リンク表示復活

3.14　まとめ

以上、Wikipedia で半自動着色読書（popilizer）のチュートリアルでした。

popilizer は HTML のテキスト要素を処理しているだけなので、Wikipedia に限らず各サイトで利用可能です。レイアウト崩れとか構文解析ミスとかまだまだ改善点が多いですが、是非、普段見ているサイトで遊んでみてください。

日本語だけでなく、英語のサイトでも動きます。HTML を使っている系のサービスも試すと面白いかもです。一例としては技術書を HTML 形式で提供しているサービスがあったりします[*13]

pdf2htmlex を使える人は PDF を HTML に変えてローカルファイルを KiwiBrowser で読み込ませれば popilizer が使えます。半自動読書が有効なのは、教科書みたいなかっちりした文章です[*14]。特に、「重要語を表題から拾える、その拾った重要語が本文中でもそのまま出現してくれる」というのがかなり効いてきます。

今回ご紹介したやり方は、大概の文書に通用するやり方という手応えを得ていますが、恐らくもっと良い戦略があるはずです。他にも、文章の種類ごとに着色戦略の各論があったりします[*15]。プリセット語もそれぞれ変わってきます。それらについては、またの機会にご紹介します。よろしくおねがいします。

[*13] BOOK TECH 社 (https://book-tech.com/)

[*14] 逆に Twitter みたいなラフな会話は今の所苦手です。異字同義語、カテゴリ分類、それも読者がその時に持っている観点の反映方法が肝になってきそうです。区別したい観点、重要語は文脈により異なります。

[*15] 上で述べた Twitter の他、2ch などの掲示板、青空文庫等の小説、ニュースサイト、個人ブログ、官報、プレスリリース、契約書などの同意書、装置のマニュアル、ショッピングサイト、商品のレビューなど

あとがき

感想と今後の展開

以上、駆け足でしたが半自動着色読書技術の最新の状況でした。昨年は PDF リーダーで半自動着色読書をするというものでしたが、今年はブラウザに展開できたので、もともとある洗練された GUI（Chromium）を流用できたし、プラットフォームの制約も緩和されました。今回は Android 端末のみでしたが、他のプラットフォームにも展開したいです。Chrome ストア登録や iOS 対応です。どちらも実機では既に動いています。ただストア申請の際には審査を受けることになるので、完璧なものを用意してから臨みたいです。現状はレイアウト崩れ、着色対象箇所の効率的な特定、構文解析の精度向上、などなど、課題が残っています。対応できるサイトも増やしていきたいです。商業化もゆくゆくは検討したいです。詳しい人良い知恵いただけると幸いです。

次回は、各文書における各論、popilizer の追加機能、PDF ファイルの扱い、そして構文解析の精度向上（型理論での取扱い、parser の実装 (applicative,monadic)、機械学習もやれれば)、辺りを追補して本書を完全版として頒布したいです。

謝辞

本書執筆および半自動読書のために生活を支えてくれている家族に感謝です。幸いなことに資金はそれなりにあるので、今のところは個人開発が続けられていますが家族の支えがなければ困難です。また、本書頒布の機会を与えてくださった技術書典の運営殿にも感謝です。とくにオンラインで頒布（情報発信）できるのは地方民としてはありがたい限りです。最後に本書および前著を読んでくださった皆様など、活動に興味を持ってくださっている方々に感謝です。興味がある人がいてくれている、というのはかなりのモチベーションになっています。恐らく人間は社会的動物なのでしょう。これを糧に引き続き開発を続けていきます。ありがとうございました。

著者紹介

T.Ishikawa / @polymonyrks

関数型玩具製作所 (`https://www.polymony.net`) の管理人。

表紙のどうぶつ

スローポーク（なまけもの）、Beanie Babies。

半自動読書

2021 年 7 月 10 日　技術書典 11 版 v1.0.0

著　者　T.Ishikawa / @polymonyrks
発行所　関数型玩具製作所