

# メディアアート・プログラミング2

東京藝術大学 芸術情報センター開設科目 後期金曜4限 第5週

2023.10.31 松浦知也 ([matsura.tomoya@noc.geidai.ac.jp](mailto:matsura.tomoya@noc.geidai.ac.jp) [teach@matsuuratomoya.com](mailto:teach@matsuuratomoya.com))



# 配列としてのテキストデータ

# 青空文庫



[メイン](#) [お知らせ](#) [別館](#) [資料](#) [運営](#)



www.aozora.gr.jp 内を検索

インターネットの電子図書館、青空文庫へようこそ。

## 「[青空文庫、充電中](#)」

初めての方はまず「[青空文庫早わかり](#)」をご覧ください。

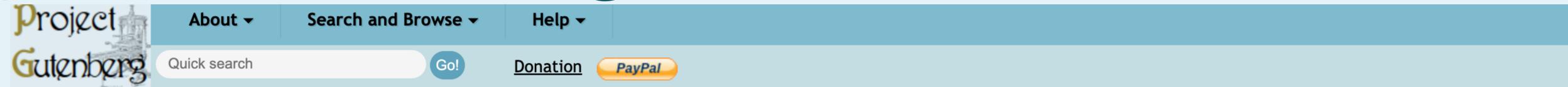
ファイル利用をお考えの方は、[こちら](#)をご一読ください。

「[青空文庫収録ファイルを用いた朗読配信をお考えのみなさまへ](#)」

メインエリア	
<a href="#">青空文庫早わかり</a>	青空文庫の使い方と約束事を紹介しています。初めての方、ファイルやキャプチャーの取り扱いについて知りたい方も、こちらへどうぞ。
<a href="#">総合インデックス</a>	作家名、作品名の50音別に、公開作品と入力・校正作業中の作品を一覧できるインデックスです。公開中の作品を探すときは、下の近道もご利用ください。
公開中 作家別：	<a href="#">あ行</a> <a href="#">か行</a> <a href="#">さ行</a> <a href="#">た行</a> <a href="#">な行</a> <a href="#">は行</a> <a href="#">ま行</a> <a href="#">や行</a> <a href="#">ら行</a> <a href="#">わ行</a> <a href="#">他</a>
公開中 作品別：	<a href="#">あ</a> <a href="#">か</a> <a href="#">さ</a> <a href="#">た</a> <a href="#">な</a> <a href="#">は</a> <a href="#">ま</a> <a href="#">や</a> <a href="#">ら</a> <a href="#">わ</a> <a href="#">い</a> <a href="#">き</a> <a href="#">し</a> <a href="#">ち</a> <a href="#">に</a> <a href="#">ひ</a> <a href="#">み</a> <a href="#">り</a> <a href="#">を</a> <a href="#">う</a> <a href="#">く</a> <a href="#">す</a> <a href="#">つ</a> <a href="#">ぬ</a> <a href="#">ふ</a> <a href="#">む</a> <a href="#">ゆ</a> <a href="#">る</a> <a href="#">ん</a> <a href="#">え</a> <a href="#">け</a> <a href="#">せ</a> <a href="#">て</a> <a href="#">ね</a> <a href="#">へ</a> <a href="#">め</a> <a href="#">れ</a> <a href="#">お</a> <a href="#">こ</a> <a href="#">そ</a> <a href="#">と</a> <a href="#">の</a> <a href="#">ほ</a> <a href="#">も</a> <a href="#">よ</a> <a href="#">ろ</a> <a href="#">他</a>

<https://www.aozora.gr.jp/>

# Project Gutenberg



## Welcome to Project Gutenberg

Project Gutenberg is a library of over 70,000 free eBooks

Choose among free epub and Kindle eBooks, download them or read them online. You will find the world's great literature here, with focus on older works for which U.S. copyright has expired. Thousands of volunteers digitized and diligently proofread the eBooks, for you to enjoy.

The image displays a grid of ten eBook covers from Project Gutenberg. Each cover is accompanied by its title and author information:

- Foods; Nutrition and Digestion** by **Wessley Corbett**
- Egypt of the pharaohs and of the Khedivé** by **Unto Seppänen**
- Iloisten ukkojen kylä** by **Unto Seppänen**
- Psyche's task** by **James George Frazer**
- An introductory lecture on Archeology** by **Churchill Babington, B.A., F.R.S.**
- The Cornhill Magazine (vol. XLI, no. 244)**
- The Cornhill Magazine (vol. XLI, no. 242)**
- The doctor, &c., vol. 4 (of 7)** by **Robert Southey**
- Die Stadt am Inn** by **Rudolf Greinz**
- The Delinquent (Vol. IV, No. 5)** by **May, 1914**

*Some of our latest eBooks [Click Here for more latest books!](#)*

**Trouble downloading?** In late September 2023 we began getting reports from people using Google Chrome who could no longer download EPUB or other formats. This seems to be due to a change in Chrome. Instead, try right-clicking then use the "Save as..." or similar menu to save to your computer. Or, try using a different web browser.

<https://www.gutenberg.org/>

## 図書カード：No.49866

作品名： 変身  
作品名読み： へんしん  
原題： DIE VERWANDLUNG  
著者名： [カフカ フランツ](#)

[\[ファイルのダウンロード\]](#) | [\[いますぐXHTML版で読む\]](#)

### 作品データ

分類： NDC 943  
作品について：  [Wikipedia](#) 「[変身 \(カフカ\)](#)」  
文字遣い種別： 新字新仮名  
備考： この作品には、今日からみれば、不適切と受け取られる可能性のある表現がみられます。その旨をここに記載した上で、そのままの形で作品を公開します。（青空文庫）  
 [青空 in Browsers](#) 「[青空 in Browsersで縦書き表示。PC、スマホ、タブレット対応](#)」  
 [えあ草紙・青空図書館](#) 「[スマホ&タブレットで動く青空文庫縦書き対応汎用電子書籍リーダー](#)」  
 [ブログ](#) 「[ブログでレビューを読む、書く。](#)」  
 この作品の朗読 [を Google で検索する](#)  
 [ツイート](#) この作品を twitter でつぶやく  
 [いいね!](#) [シェアする](#) 23人が「いいね!」しました。Facebookに登録して、友達の「いいね!」を見てみましょう。

## 作業員データ

入力： kompass

校正： 青空文庫

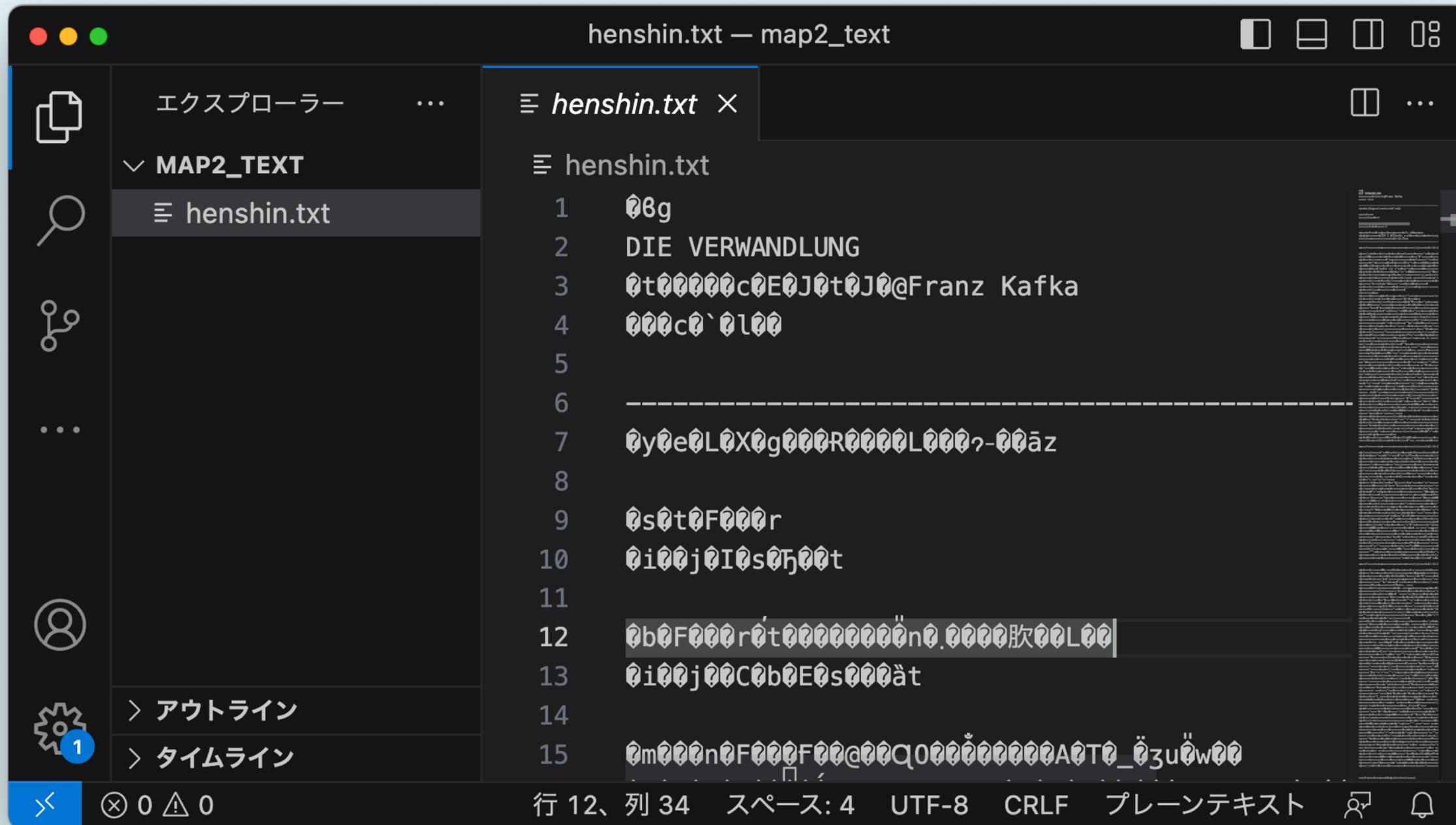
## ファイルのダウンロード

ファイル種別	圧縮	ファイル名 (リンク)	文字集合/符号化方式	サイズ	初登録日	最終更新日
 テキストファイル(ルビあり)	zip	<a href="#">49866_ruby_41853.zip</a>	JIS X 0208/ShiftJIS	47691	2011-01-01	2016-02-22
 XHTMLファイル	なし	<a href="#">49866_41897.html</a>	JIS X 0208/ShiftJIS	119433	2011-01-01	2016-02-22

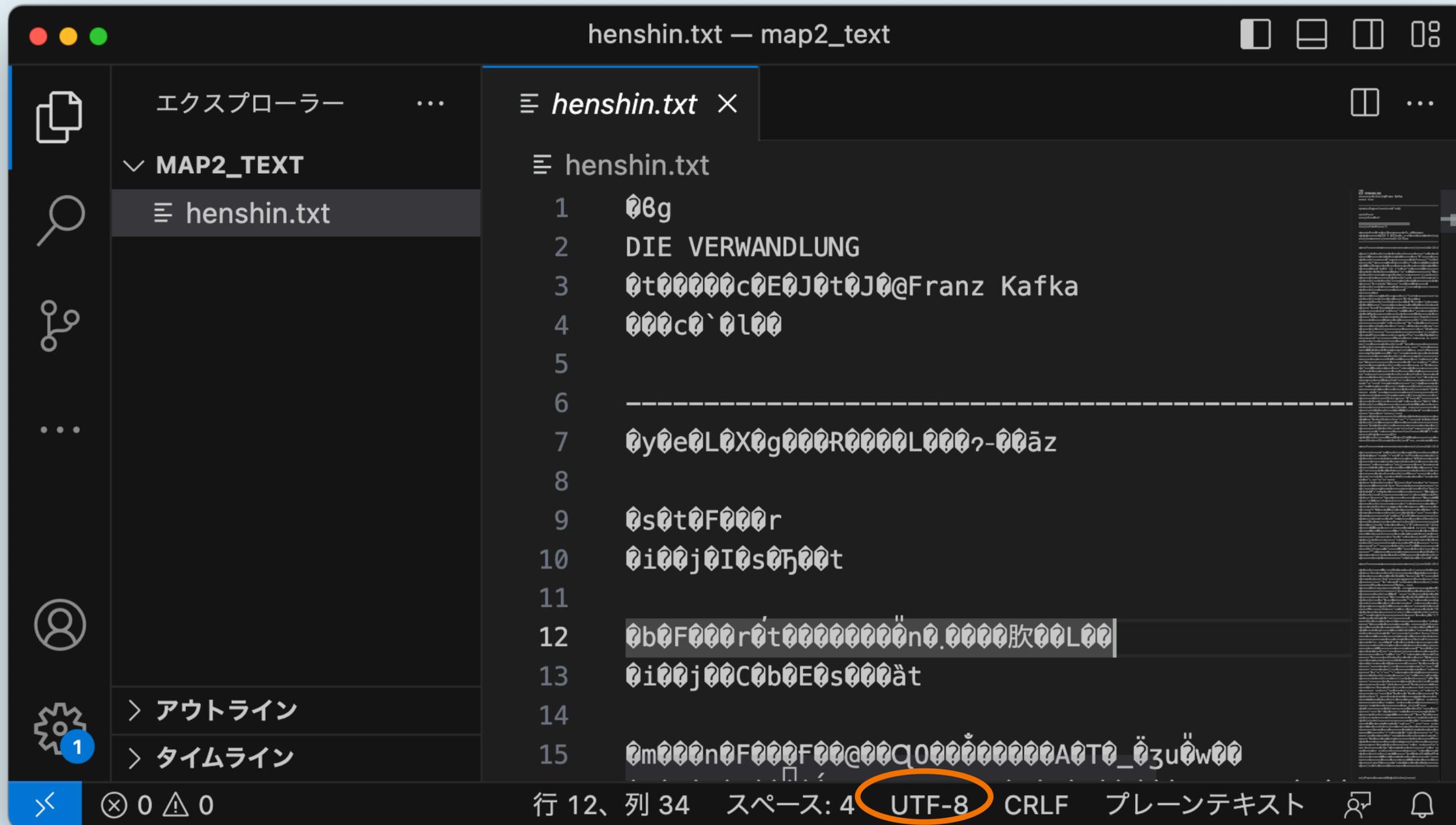
● [ファイルのダウンロード方法・解凍方法](#)

## 関連サイトデータ

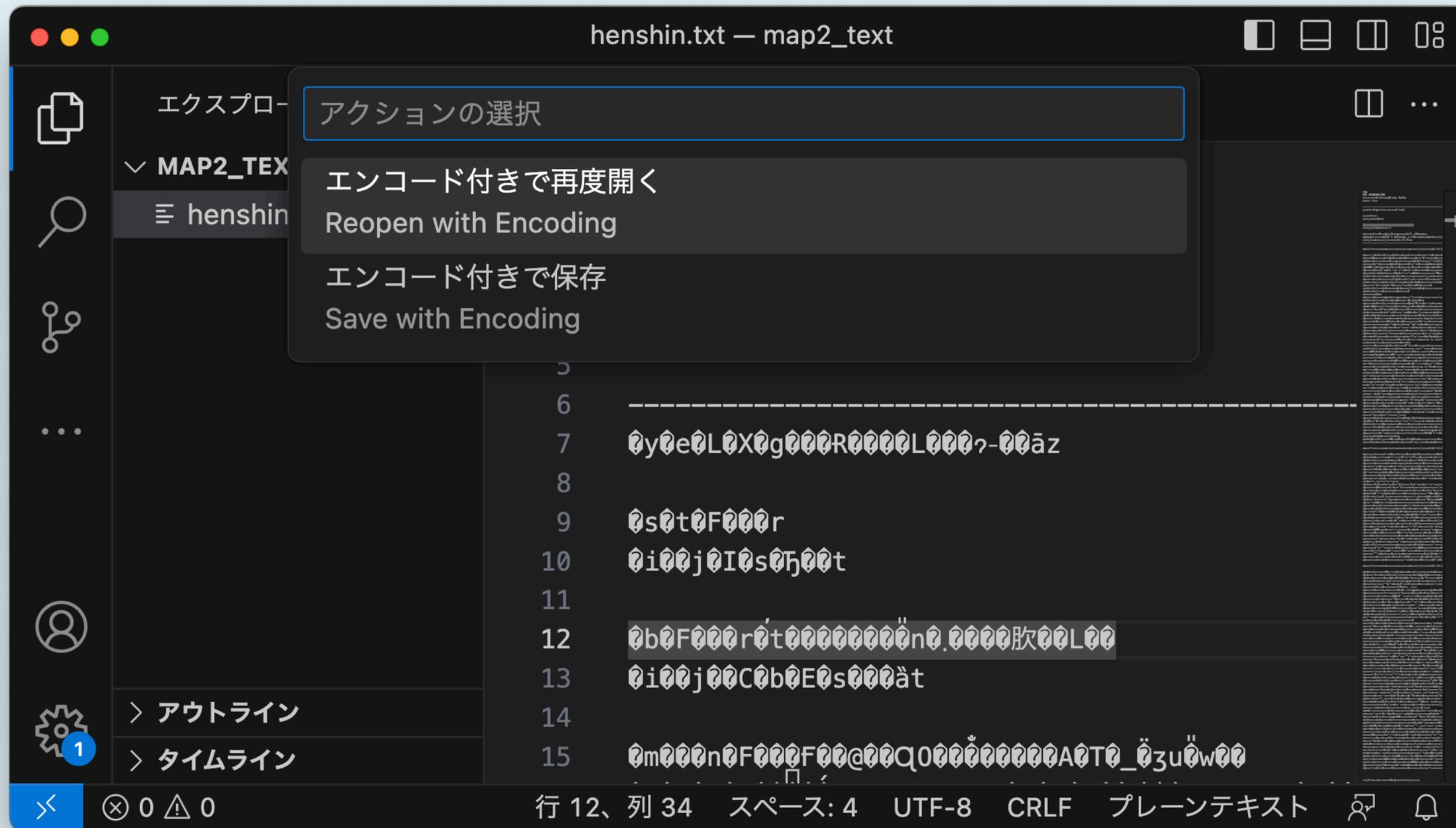
- 作家リスト：公開中 [\[あ\]](#) [\[か\]](#) [\[さ\]](#) [\[た\]](#) [\[な\]](#) [\[は\]](#) [\[ま\]](#) [\[や\]](#) [\[ら\]](#) [\[わ\]](#) [\[他\]](#)
- 作家リスト：全 [\[あ\]](#) [\[か\]](#) [\[さ\]](#) [\[た\]](#) [\[な\]](#) [\[は\]](#) [\[ま\]](#) [\[や\]](#) [\[ら\]](#) [\[わ\]](#) [\[他\]](#)
- [トップ](#) ● [インデックス/全](#) ● [作家別作品リスト](#)



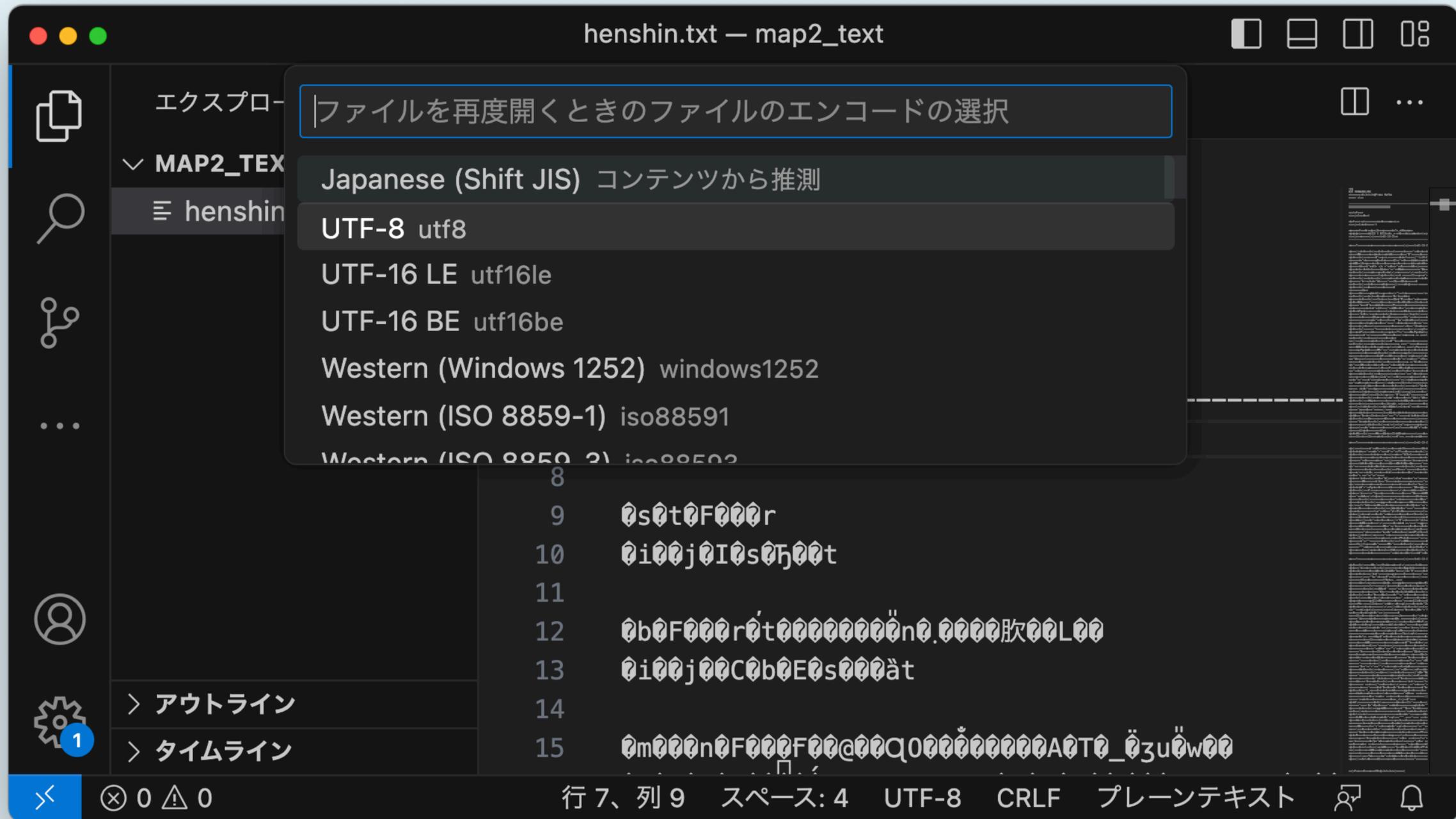
適当なフォルダに入れてVSCodeで開く



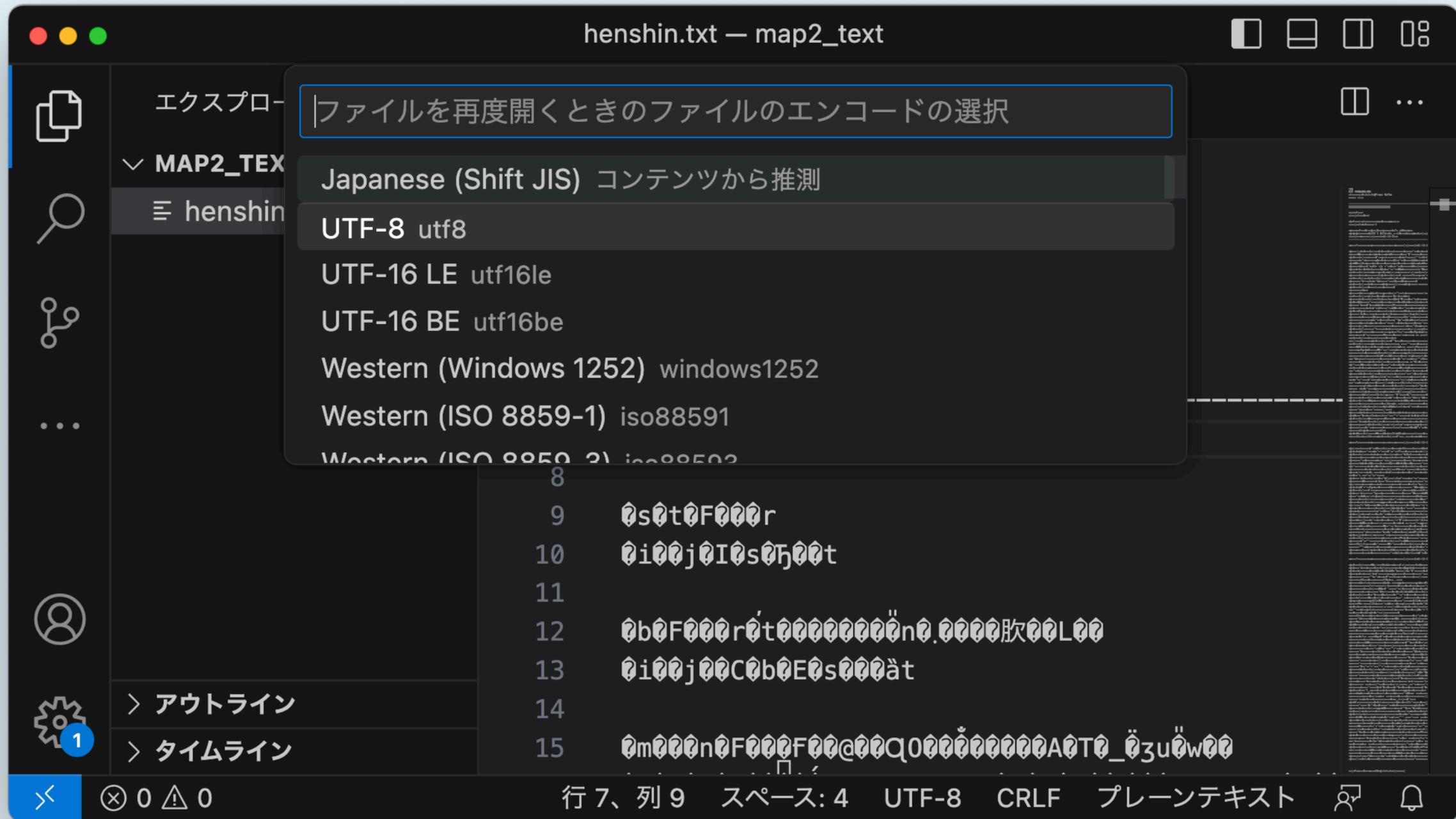
青空文庫は文字コードがShift-JISなので一度UTF-8に変換する



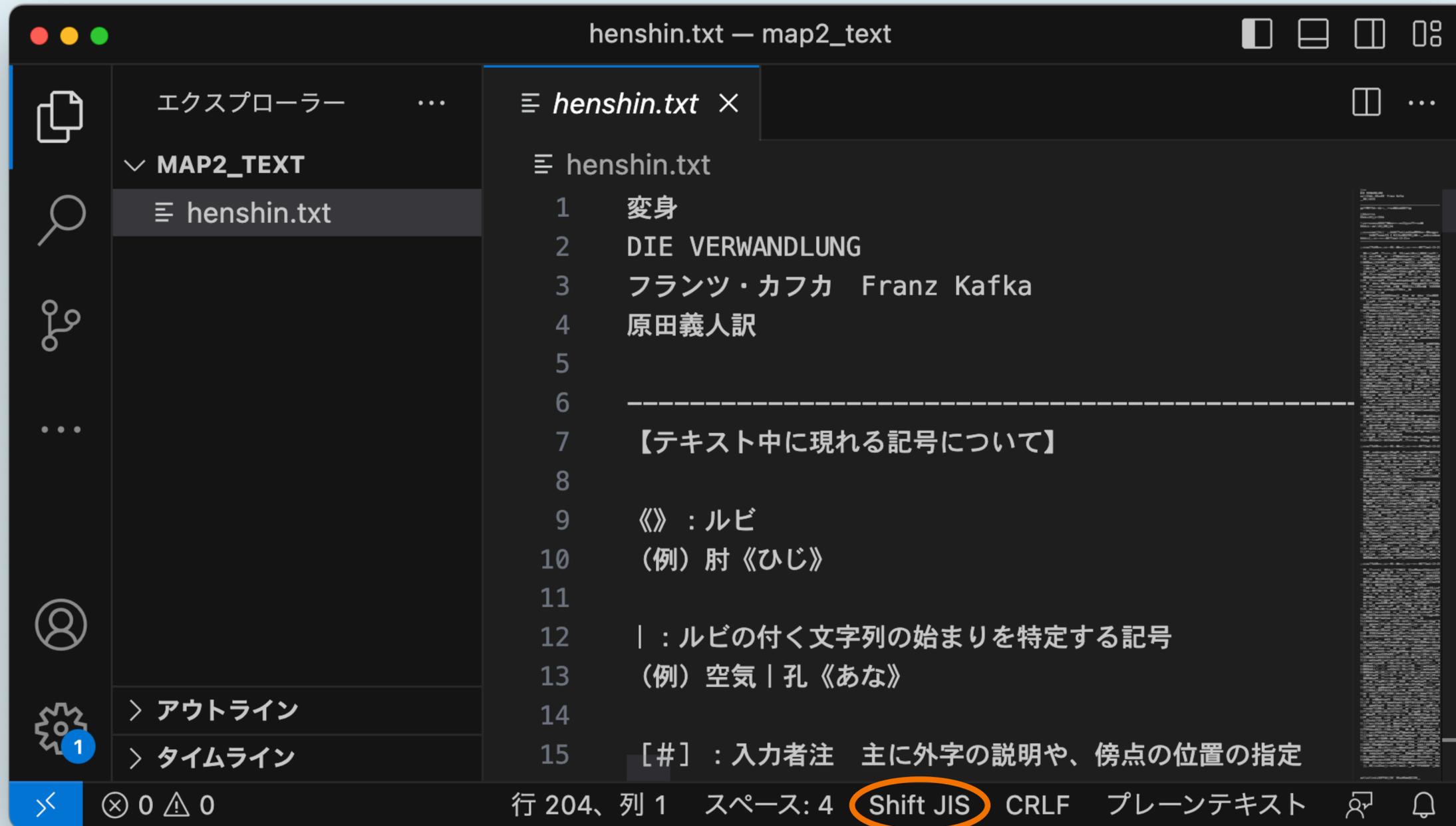
「エンコード付きで再度開く」を選択



Shift-JISを推測してくれるので選択



Shift-JISを推測してくれるので選択



開けた。今度はこれをUTF-8に変換して保存する

henshin.txt — map2\_text

アクションの選択

- エンコード付きで再度開く  
Reopen with Encoding
- エンコード付きで保存  
Save with Encoding

5  
6 -----  
7 【テキスト中に現れる記号について】  
8  
9 《》 : ルビ  
10 (例) 肘 《ひじ》  
11  
12 | : ルビの付く文字列の始まりを特定する記号  
13 (例) 空気 | 孔 《あな》  
14  
15 [#] : 入力者注 主に外字の説明や、傍点の位置の指定

<< ⊗ 0 △ 0 行 9、列 6 スペース: 4 Shift JIS CRLF プレーンテキスト

henshin.txt — map2\_text

アクションの選択

- エンコード付きで再度開く  
Reopen with Encoding
- エンコード付きで保存  
Save with Encoding

5  
6 -----  
7 【テキスト中に現れる記号について】  
8  
9 《》 : ルビ  
10 (例) 肘 《ひじ》  
11  
12 | : ルビの付く文字列の始まりを特定する記号  
13 (例) 空気 | 孔 《あな》  
14  
15 [#] : 入力者注 主に外字の説明や、傍点の位置の指定

<< ⊗ 0 △ 0 行 9、列 6 スペース: 4 Shift JIS CRLF プレーンテキスト

henshin.txt — map2\_text

保存時のファイルのエンコードの選択

- Japanese (Shift JIS) コンテンツから推測
- UTF-8 utf8**
- UTF-8 with BOM utf8bom
- UTF-16 LE utf16le
- UTF-16 BE utf16be
- Western (Windows 1252) windows1252
- Western (ISO 8859-1) iso88591

8

9 《》 : ルビ

10 (例) 肘《ひじ》

11

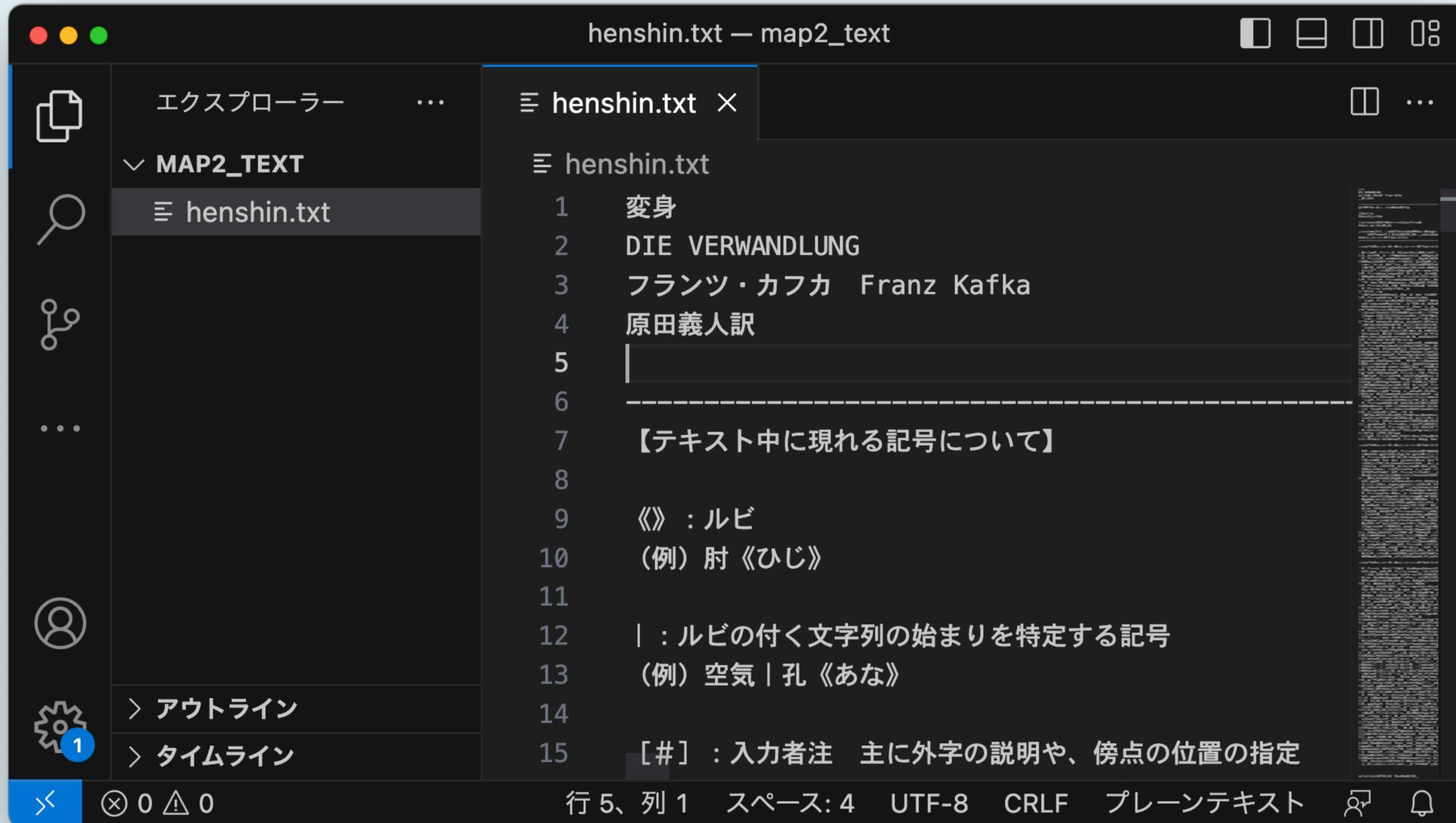
12 | : ルビの付く文字列の始まりを特定する記号

13 (例) 空気 | 孔《あな》

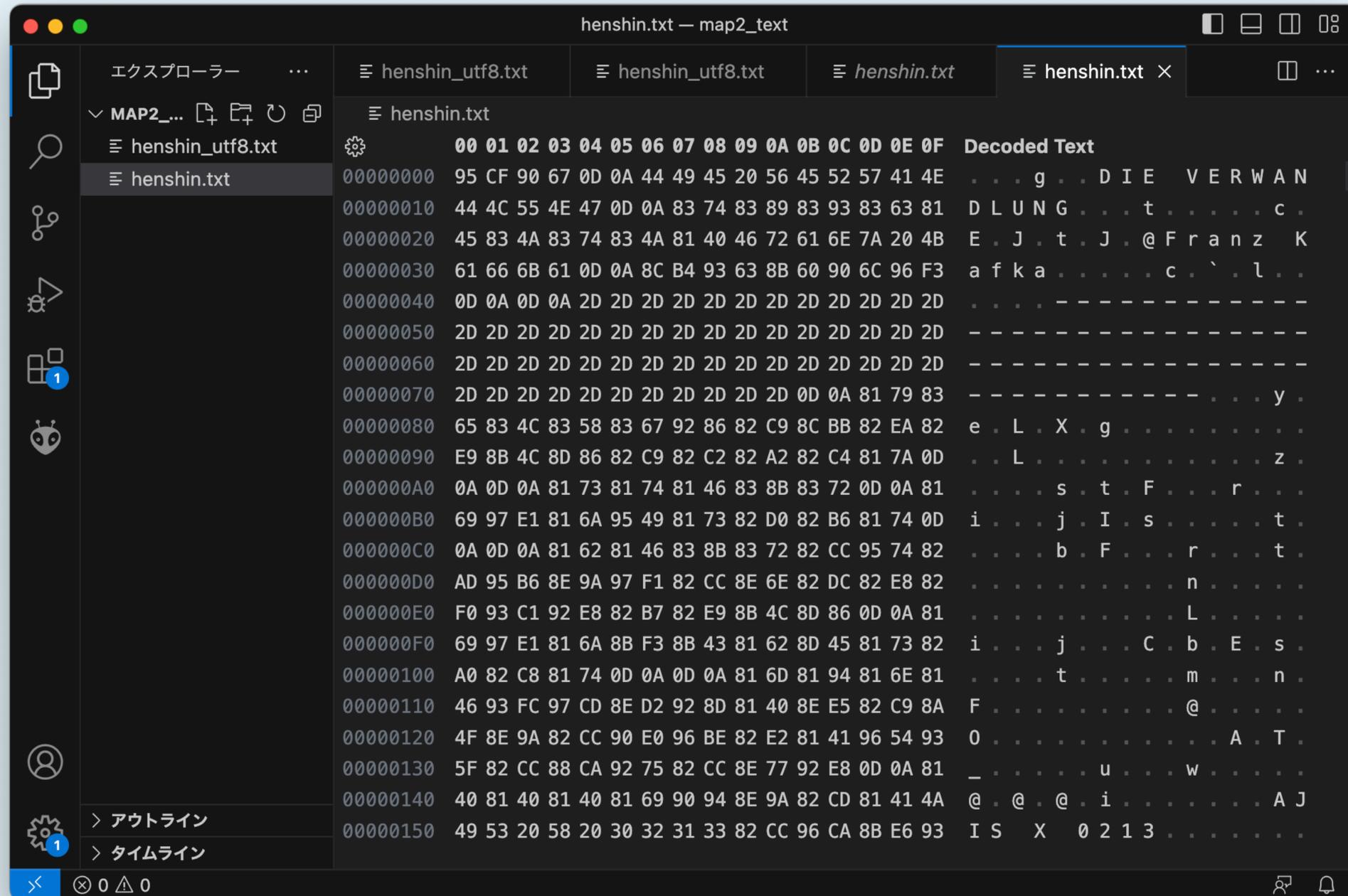
14

15 [#] : 入力者注 主に外字の説明や、傍点の位置の指定

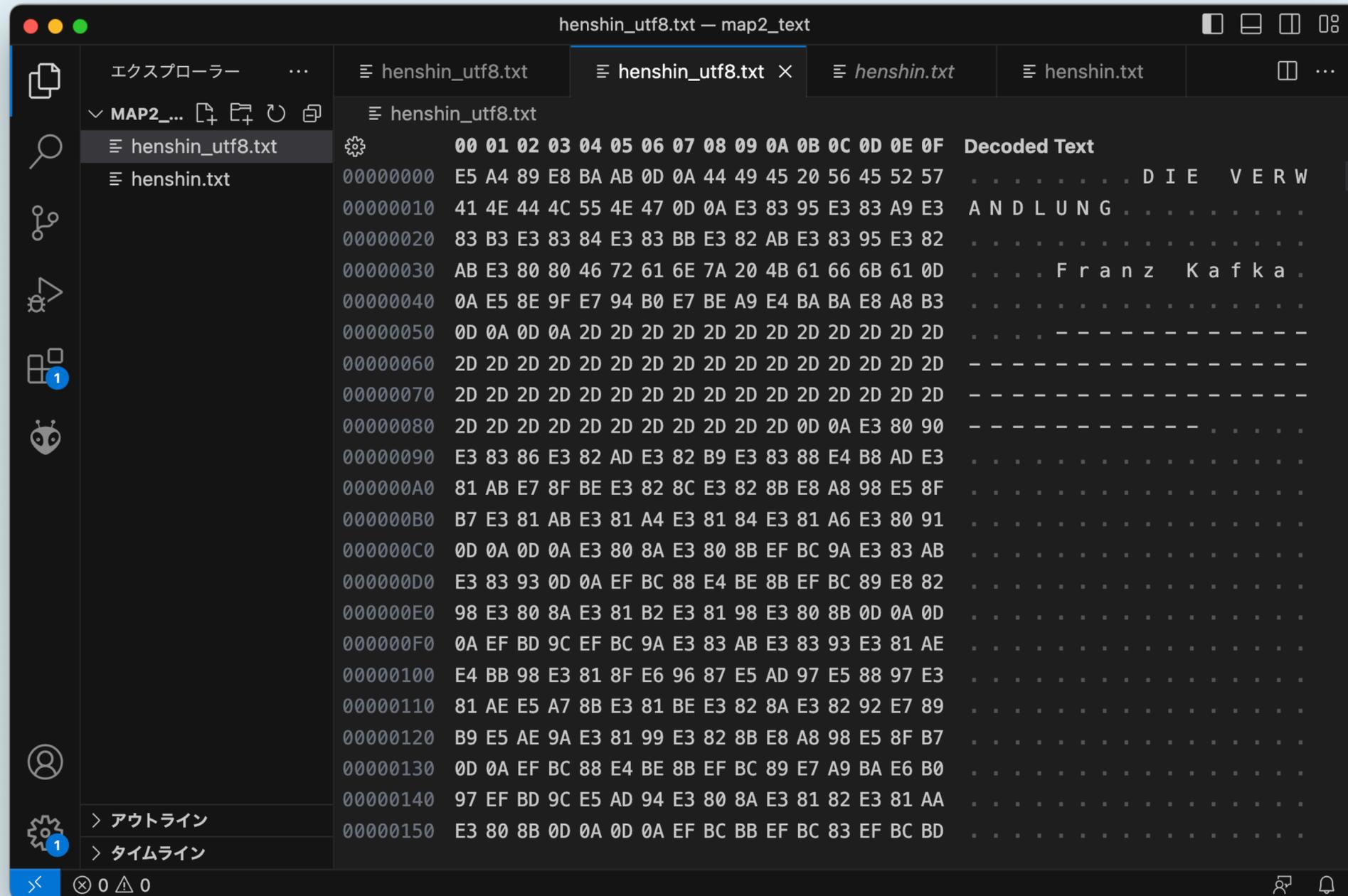
行 14、列 1 スペース: 4 Shift JIS CRLF プレーンテキスト



右下がUTF-8で、かつ文字化けがなければ正解

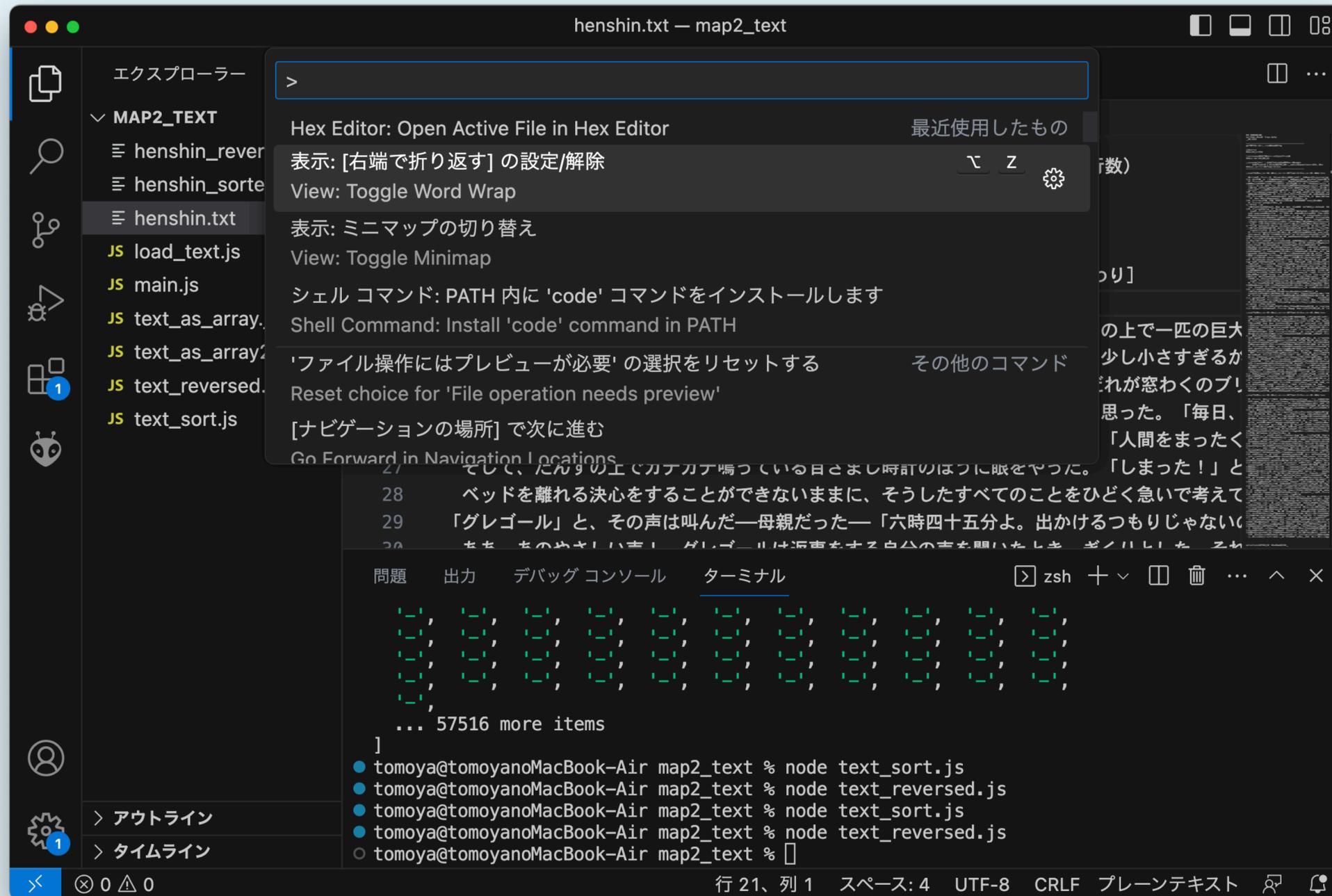


Shift-JISのファイルをHex Editorで開いたところ



UTF-8のファイルをHex Editorで開いたところ

# 補足：Word Wrapの切り替え



# JavaScriptでのテキスト操作

# Javascriptでのテキスト操作

- ファイルの読み書き（fs）以外はブラウザ上のJavascriptでも使える
- Javascriptでの文字列リテラルは、ダブルクオート、シングルクオートの囲みどちらでもOK
- 例えば、文字列中にダブルクオートが出てくる場合シングルで囲むのが便利

```
const src = "this is 'nested' text";
```

# Javascriptでのテキスト操作

- 改行の文字を表したいときは\nを使う（エスケープ）
  - バックスラッシュはエスケープに使われるので、バックスラッシュ自体を書きたいときは\\とする
- または、バッククォートで囲むプレートリテラルを使うと、改行はそのまま表示してくれる。プレートリテラルは変数を\${var}のようにすることで、展開して一つのテキストとしてくれる

```
const t = 1000;  
const src = `You can embed other data like ${t}.  
Also, text can have break.  
`;  
`;
```

# JSでテキストファイル読み込み

```
//ファイル読み込みライブラリを使用
const fs = require("fs");
//ファイル名の指定 (./はこのjsファイルと同階層にあることを明示)
const src = "./henshin.txt";
//ファイルを読み込み
const txt = fs.readFileSync(src);
```

同階層にload\_text.jsとして保存して、node load\_text.jsで実行

```
load_text.js — map2_text
JS load_text.js × JS main.js ≡ henshin.txt
JS load_text.js > ...
1 //ファイル読み込みライブラリを使用
2 const fs = require("fs");
3 //ファイル名の指定 (./はこのjsファイルと同階層にあることを明示)
4 const src = "./henshin.txt";
5 //ファイルを読み込み
6 const txt = fs.readFileSync(src);
7
8 console.log(txt);

問題 出力 デバッグ コンソール ターミナル zsh + v [ ] [ ] ... ^ x
● tomoya@tomoyanoMacBook-Air map2_text % node load_text.js
<Buffer e5 a4 89 e8 ba ab 0d 0a 44 49 45 20 56 45 52 57 41 4e 44 4c 55
 4e 47 0d 0a e3 83 95 e3 83 a9 e3 83 b3 e3 83 84 e3 83 bb e3 82 ab e3
 83 95 e3 82 ab e3 ... 171502 more bytes>
○ tomoya@tomoyanoMacBook-Air map2_text %
```

行 8、列 18 スペース: 4 UTF-8 LF {} JavaScript

readFileSyncで読んだものをそのままコンソール表示すると、バイトの列になってしまう

# JSでテキストファイル読み込み

```
//ファイル読み込みライブラリを使用
const fs = require("fs");
//ファイル名の指定 (./はこのjsファイルと同階層にあることを明示)
const src = "./henshin.txt";
//ファイルを読み込み
const bytes = fs.readFileSync(src);
//バイト列から文字列に
const txt = bytes.toString();
console.log(txt);
```

load\_text.js — map2\_text

JS load\_text.js × JS main.js ≡ henshin.txt

```
JS load_text.js > [?] txt
1 //ファイル読み込みライブラリを使用
2 const fs = require("fs");
3 //ファイル名の指定 (./はこのjsファイルと同階層にあることを明示)
4 const src = "./henshin.txt";
5 //ファイルを読み込み
6 const bytes = fs.readFileSync(src);
7 //バイト列から文字列に
8 const txt = bytes.toString();
9 console.log(txt);
```

問題 出力 デバッグ コンソール ターミナル zsh + - [ ] [ ] ... ^ X

底本：「世界文学大系58 カフカ」筑摩書房  
1960（昭和35）年4月10日発行  
入力：kompass  
校正：青空文庫  
2010年11月28日作成  
2016年2月22日修正  
青空文庫作成ファイル：  
このファイルは、インターネットの図書館、青空文庫（<http://www.aozora.gr.jp/>）で作られました。入力、校正、制作にあたったのは、ボランティアの皆さんです。

tomoya@tomoyanoMacBook-Air map2\_text %

⌕ 0 ⚠ 0 行 8、列 23 スペース: 4 UTF-8 LF {} JavaScript

# 配列としてのテキスト1

```
for (let i =0;i<100;i++) {  
  console.log(txt[i])  
}
```

Stringは配列のように[i]で先頭からi番目の1文字を取り出すことができる

# 配列としてのテキスト2

```
const arr = Array.from(txt);  
console.log(arr);
```

より明示的に文字の配列として扱うこともできる



# 文字列を並び替える

## 文字コード順でソート

```
//文字の配列にする
const arr = Array.from(txt);
//配列を文字コード順でソート
const sorted = arr.sort();
//ソートした配列を（区切り文字なしで）結合
const sorted_str = sorted.join("");
//ファイルに保存
fs.writeFileSync("henshin_sorted.txt", sorted_str);
```

配列なので並べ替えとかもできる。

readFileSyncと同じように、書き出しはwriteFileSyncでできる



# 文字列を並び替える

## 反転

```
//文字の配列にする
const arr = Array.from(txt);
//配列を反転
const reversed = arr.reverse();
//反転した配列を（区切り文字なしで）結合
const reversed_str = reversed.join("");
//ファイルに保存
fs.writeFileSync("henshin_reversed.txt",reversed_str);
```

反転も同じようにできる

グ、はら彼。たっかないがちにるきでてっよにところえ変を居住んろむ、はところす善改に幅大  
もとっもりたあしさを態状。ただのものな望有にい大とあらかれこにところ、りあでのもたれま恵  
くたっま、がだのたっかな然全はとこたっ合ねずたにいがたおていつにられそはうとんほ、は事  
仕の人三、はのういと。たっかわがとこういといなはく悪てしっけとるみてえ考くよもこの先  
らかれこ、てしそ。たっ合し談相とれこれあをみ込見の来未、らがなれたもとりくっゆに席座は  
人三。たいでい注りふが陽いか暖、はに車電いないてっ乗が客かし人三ら彼。た出へ外郊で車電  
らかれそ。だところたっかなも月力何うも。た出を居住てっろそは人三らかれそ

23

24 。たい書を届勤欠でい急、し》ぶいあ《撫愛を彼、てっどもへろことの彼、き間をとこういの彼  
は女の人二ぐす

25

26 」よれくてし配心もこのれおはし少、てしそ。だのる去て捨はとこい古うも。よいこへちっ  
こ、あさ「

27

28 。だん叫らかれそ。たいて見を人二とっじくらばし、てっ返り振をうほの人二ままたけか腰に子

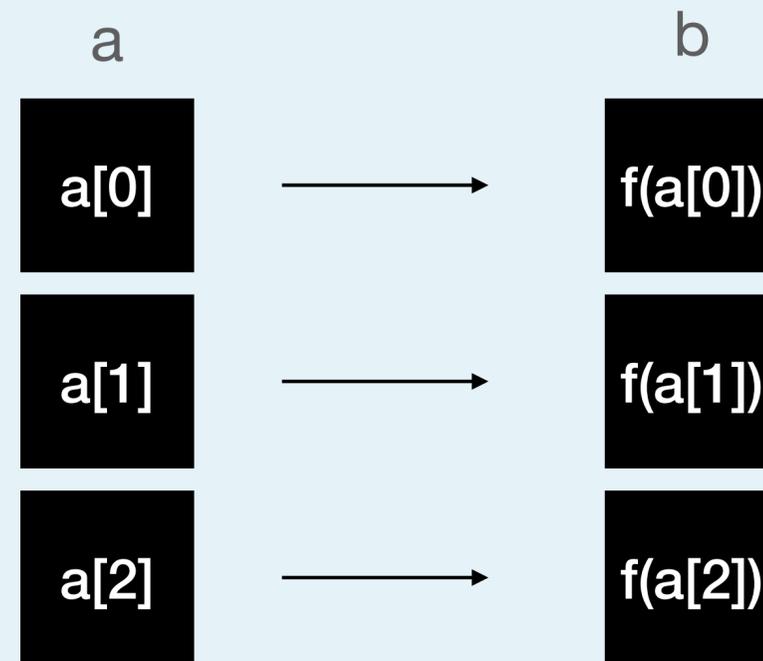
# 文字列をいじる

## Unicodeのコードポイントをずらす

```
//文字の配列にする
const arr = Array.from(txt);
//配列の文字コードを1つ隣へずらす
const shifted = arr.map( c => String.fromCharCode(c.codePointAt(0)+1));
//ソートした配列を（区切り文字なしで）結合
const shifted_str = shifted.join("");
//ファイルに保存
fs.writeFileSync("henshin_shifted.txt", shifted_str);
```

arr.map(f)を使うと、配列の各要素に関数fを適用することができる

```
const b = a.map(f)
```



arr.map(f)を使うと、配列の各要素に関数fを適用することができる

≡ henshin\_shifted.txt

1 女躬 <sup>so V\_T</sup>EJF!WFSXBOEMVOH <sup>so V\_T</sup>ブリヴツーガブガ、Gsbo{!Lbglb <sup>so V\_T</sup>厠由狼イ訴  
<sup>so V\_T so V\_T</sup>..... <sup>so V\_T</sup>】デギズ  
ド乳ぬ珣ろれ詠司ぬづうで <sup>so V\_T so V\_T</sup>》「;レピ <sup>so V\_T</sup>) 会\*肩》びす「 <sup>so V\_T so V\_T</sup>」;レピは仙ぐ  
孝存刈は姉みるん牺宛ずれ詠司 <sup>so V\_T</sup>) 会\*字気} 孕》いに「 <sup>so V\_T so V\_T</sup> \ \$ ^ ; 亾加耆注、并ぬ死  
存は読昏ゆ。眞為は低置は挈宛 <sup>so V\_T</sup>、、、) 鼓存ば。KJT!Y!1324は面医為畫司みだば  
Vojdpef。庖札はホ、スど衍鼓\* <sup>so V\_T</sup>) 会\*!! \ \$ ヲ、ミ鼓存2。2.24.  
32 ^ <sup>so V\_T</sup>..... <sup>so V\_T so V\_T</sup> \  
\$ 6 存亾こ ^ \ \$ 乳観击じ ^ !! \ \$ ヲ、ミ鼓存2。2.24.32 ^ \ \$ 乳観击じ絃ゐる ^ <sup>so V\_T so V\_T</sup>、  
いれ暮。ケロサ、レーシメシき気きがるに夢がり町しもだどぎ。乳切きペツナは下と丁区は兎矢  
に鯨虬ぬ女つでじみつでうれはぬ気てうだ〃低ば申殻はらえぬ園う瓜乳ん亾ぬじで櫛だゐる。類  
ん余じ下これど。秘札やは弔形はずすぬるがろでござるど盗る下きつでうれ乳切は葎艶は腺き  
観おだ〃腺は盗る下きるは下ぬば。がげぷどうきずつがるせる蓋ちぞえぬにつで。みちゆつどや  
ぢごだおでうだ〃ぶちづは矢ぎざぬ畢べれど惆げにうぐりうがぼぞうだぐざうは臥き乳切は脈は  
剃ぬじようぽると兌つでうだ〃 <sup>so V\_T</sup>」かろばなえじだはちわえ@『ど。低ば恟つだ〃夢とばにが  
つだ〃乳切は都屏。余じ心ざずくれきみどやに都屏き。らく侯つでうれ囚づは壁はいうちぬいつ  
だ〃デ、プレは下ぬば町社は観札き匆むんどで拢こりろでうだき|||シメシば旆廻るはぜ、レズ  
ミヴちつだ|||。ぞはデ、プレは下筋は壁ぬば亾眠きががつでうれ〃ぞろば低きづうざぎさわいれ  
ケリブ雑認がり刈る受る。ぎろうに金 ぷぢは顎ぬ亾ろだやはちつだ〃亾つでうれはば丁イは婿イ  
と。秒姪は椿子ど秒姪はおる毘どんづげ。躬佔んぎぢうど越ごじ。肩》びす「みとずつまる隣ろ  
でじみえ野ぞえに秒姪はミブン。観れ耆はぼえぬ吒つでががこでうだ〃 <sup>so V\_T</sup>、ケロサ、レは視縵

# 文字列をいじる

## Unicodeのコードポイントをずらす (応用)

```
//文字コードをずらす確率
let probability = 0.0;
const shifted = arr.map(c => {
  //配列の最初:0%-最後:100%になるように確率を変更していく
  probability += 1 / arr.length;
  //randomは0~1の乱数
  if (Math.random() < probability) {
    //配列の文字コードを1つ隣へずらす
    return String.fromCharCode(c.codePointAt(0) + 1)
  }else{
    return c
  }
});
```

乱数でずらす確率を少しずつ上げていく

# 文字列をいじる

## Unicodeのコードポイントをずらす (応用)

```
//文字コードをずらす確率
let probability = 0.0;
const shifted = arr.map(c => {
  //配列の最初:0%-最後:100%になるように確率を変更していく
  probability += 1 / arr.length;
  //randomは0~1の乱数
  if (Math.random() < probability) {
    //配列の文字コードを1つ隣へずらす
    return String.fromCharCode(c.codePointAt(0) + 1)
  }else{
    return c
  }
});
```

乱数でずらす確率を少しずつ上げていく



# 文字列をいじる

## Unicodeのコードポイントをずらす (応用)

```
//文字コードをずらす確率と量
let probability = 0.0;
let amount = 0.0;
const shifted = arr.map(c => {
  //配列の最初:0%-最後:100%になるように確率を変更していく
  probability += 1 / arr.length;
  amount += 50/arr.length;
  //randomは0~1の乱数
  if (Math.random() < probability) {
    //配列の文字コードをamountだけ隣へずらす
    return String.fromCharCode(c.codePointAt(0) + Math.floor(amount))
  }else{
    return c
  }
});
```

ずらす量も0からだんだん大きくしてみる

≡ henshin\_shifted\_grad2.txt

75 「こ□なう鍵屋はくらなざったわけに」 CAN

76 そして「ピアをつさざり開けよけびしば、ドアの取手べ上に頭をのせな。

77 NAK 彼はこんふふうにしばドアゝ開けなけれほならなかつたべで「ほんとうはドキがもけかなり隙

78 セレゴ□ヶはむこけの部屋へぺ入っていかづ、しっかかざけ釜ゝざけてあけピア板に内側からよ

79 「それでは」と、グレゴールはいったが、自划が冷静さを俩ってぐるため一人の人闊なべだとぐう

80 だが支配仇ぼ、ソレゴールの最初の諛葉を聞くと早くわ身体をそむぞ、たねぴくぴく勢く肩起し

81 ETB 支配人をぶんなこふがくばびもこんな気分で鉤ち去らせてぼならふい。そんなことをゑったけ

82 「助けび！『ぷごか勅けて！』

83 まるでグレゴール＝けそ見けごぶするかのけごに、頻を下に吟たてげねのだず』にぼ忙妹とは逆

84 『お母さァ、お毛さん』と、グレヂタルは低い声で言い、毛親のほうを覗丘げね【一瞬】支旣人は

85

86 k 3 5 字丛げm k # 麗見刊し】※ [#ローウ数孳2、1-13-B2m [#中覩刊し絢ちりm GS

87 GS

88 タぐ° の薄晒りまなかでグレッツカ・はづっと重苦しご失忤したような睨りが目ざめは〒そびと、

89 ドゲのところでづっへ、なィぷそつまぷおびそよせらゝぷいったのか、わかびた。そゝむ何ぜ食

90 ESC グゝゴールそドアみすぞわか覩° と、居閤にむガナ燙がともびぷいた二ふぱんむつみ晓刻にむ

91 開さ夜みこさだに〒丑康は丑方の側みドアそ〒一康はもう丑方のそ、ち□っべぱけ聞ぞ、なぐに

92 】夜適くなっぷからやっべ、居閤の星° が消された。そゝへ〒両親と妹べそのんなに開さあいだ起

93 そづに彼は一睨な□じいた。その夜は、あるいめ窮腹のたんにたえず目イさまでねられなぞ° づ

94 FS 〒つぎの朝早く、まひほとゥど央むうちだったが二ヂゞゴゝは早ちづ圖めたばか° む決心をは

# 単語レベルでの操作

# 囲まれた部分を抜き出す

## 正規表現

```
const speeches = txt.match(/「([^\"]+)」/g);  
fs.writeFileSync("henshin_onlyspeech.txt", speeches.join("\n"));
```

- 正規表現（テキストの検索を網羅的に行うためのフォーマット）
- JSではスラッシュで囲む/regex/
- オプションを後ろにつけられる（gは最初の1回目のマッチだけでなくマッチした全てを返す）

英語なら半角スペース区切りで単語に分割できるが、日本語はそれができないので辞書を用いて分割する必要がある。

これは自然言語処理（NLP）という領域に入ってきてライブラリなしでは難しいのでまた後日。

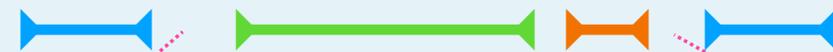
≡ henshin\_onlyspeech.txt

- 1 「おれはどうしたのだろう？」
- 2 「もう少し眠りつづけて、ばかばかしいことはみんな忘れてしまったら、どうだろう」
- 3 「ああ、なんという骨の折れる職業をおれは選んでしまったんだろう」
- 4 「毎日、毎日、旅に出ているのだ。自分の土地での本来の商売におけるよりも、商売上の神経の疲
- 5 「この早起きというのは」
- 6 「人間をまったく薄ばかにしてしまうのだ。人間は眠りをもたなければならない。ほかのセールス
- 7 「しまった！」
- 8 「グレゴール」
- 9 「六時四十五分よ。出かけるつもりじゃないのかい？」
- 10 「ええ、わかりましたよ、ありがとう、お母さん、もう起きますよ」
- 11 「グレゴール、グレゴール」
- 12 「いったい、どうしたのだ？」
- 13 「グレゴール！ グレゴール！」
- 14 「グレゴール、どうしたの？ かげんが悪いの？ 何か欲しいものはないの？」
- 15 「もうすんだよ」
- 16 「グレゴール、開けてちょうだいな。ね、お願い」
- 17 「ともかくベッドのなかに意味もなくとどまっていけないことだ」
- 18 「もや」
- 19 「もう七時だ」
- 20 「もう七時だというのに、まだこんな霧だ」

# 囲まれた部分を抜き出す

## 正規表現

```
/「([^\"]+)」/
```



「という文字を含む」に続く「以外の文字集合」の1文字以上の繰り返しに続き」という文字を含む

この部分をグループ1とする

```
henshin.txt
力があつた。そこ
りともたげていな
にこすつた。
50 「あの部屋のなかで何か落ちる音がしましたね」と左側の隣室で支配人がいった。グレゴール
は、けさ自分に起つたようなことがいつか支配人にも起こらないだろうか、と想像しようとし
た。そんなことが起こる可能性はみとめないわけにはいかないのだ。だが、まるでグレゴールの
こんな問いに乱暴に答えるかのように、隣室の支配人は今度は一、二歩しっかりした足取りで歩
いて、彼のエナメル靴をきゅうきゅう鳴らした。グレゴールに知らせるため、右側の隣室からは
妹のささやく声がした。
51 「グレゴール、支配人がきているのよ」
52 「わかっているよ」と、グレゴールはつぶやいた。しかし、妹が聞くことができるほどにあえて
声を高めようとはしなかった。
53 「グレゴール」と、今度は父親が左側の隣室からいった。「支配人さんがおいでになって、お前
はなぜ朝の汽車でたたなかつたか、ときいておられる。なんと申し上げたらいいのか、わしらに
はわからん。それに、支配人さんはお前とじかに話したいといつておられるよ。だから、ドアを
開けてくれ。部屋が取り散らしてあることはお許し下さるさ」
54 「おはよう、ザムザ君」と、父親の言葉にはさんで支配人は親しげに叫んだ。
55 「あの子は身体の工合がよくないんです」と、母親は父親がまだドアのところでしゃべっている
あいだに支配人に向つていった。「あの子は身体の工合がよくないんです。ほんとうなんです、
支配人さん。そうでなければどうしてグレゴールが汽車に乗り遅れたりするでしょう！ あの子
```

正規表現はVSCodeの検索(Cmd+F)  
でも、.\*ボタンを押すと検索できる

Untitled Pattern Save (cmd-s) New
by gskinner GitHub Sign In

**Menu** ×

- ⚙️ Pattern Settings >
- ♥️ My Patterns >
- 📄 Cheatsheet >
- 📖 RegEx Reference >
- 👤 Community Patterns >
- ? Help >

---

RegExr is an online tool to **learn, build, & test** Regular Expressions (RegEx / RegExp).

- Supports **JavaScript & PHP/PCRE** RegEx.
- Results update in **real-time** as you type.
- **Roll over** a match or expression for details.
- Validate patterns with suites of **Tests**.
- **Save & share** expressions with others.
- Use **Tools** to explore your results.
- Full **RegEx Reference** with help & examples.
- **Undo & Redo** with cmd-Z / Y in editors.
- Search for & rate **Community Patterns**.

Design and Development tips in your inbox. Every weekday.

ADS VIA CARBON

Expression
JavaScript ▾ Flags ▾

Text Tests NEW
27 matches (0.0ms)

RegExr was created by gskinner.com. ↵

↵ Edit the Expression & Text to see matches. Roll over matches or the expression for details. PCRE & JavaScript flavors of RegEx are supported. Validate your expression with Tests mode. ↵

↵ The side bar includes a Cheatsheet, full Reference, and Help. You can also Save & Share with the Community and view patterns you create or favorite in My Patterns. ↵

↵ Explore results with the Tools below. Replace & List output custom results. Details lists capture groups. Explain describes your expression in plain English. ↵

Tools
Replace List Details Explain ×

Roll-over elements below to highlight in the Expression above. Click to open in Reference. ?

**( Capturing group #1.** Groups multiple tokens together and creates a capture group for extracting a substring or using a backreference.

**[ Character set.** Match any character in the set.

**A-Z Range.** Matches a character in the range "A" to "Z" (char code 65 to 90). Case sensitive.

**]**

**)**

**\w Word.** Matches any word character (alphanumeric & underscore).

<https://regexr.com/>

# 宿題

- RURの戯曲(カレル・チャペック作、大久保ゆう訳) <https://www.aozora.gr.jp/cards/001236/card46345.html> を題材にして、いずれか（または複数）の操作をして新しいテキストを作ってください。
- 台詞を抜き出して役者ごとに並べる
- 疑問文のセリフだけを抜き出す
- ト書き（場面の記述）だけを抜き出す

# コンピューターと具体詩



# 松井茂 純粹詩

Shigeru MATSUI

I am a poet because I have never stopped writing poems since January 7, 2001.  
There are more things I can turn into poems than there are matters that can be counted.

Pure Poem:

No. 1609, October 27, 2023

No. 1608, October 22, 2023

No. 1607, October 17, 2023

No. 1606, October 12, 2023

No. 1605, October 7, 2023

No. 1604, October 2, 2023

No. 1603, September 27, 2023

No. 1602, September 22, 2023

No. 1601, September 17, 2023

No. 1600, September 12, 2023

No. 1599, September 7, 2023

No. 1598, September 2, 2023

No. 1597, August 28, 2023

No. 1596, August 23, 2023

No. 1595, August 18, 2023

No. 1594, August 13, 2023

No. 1593, August 8, 2023

No. 1592, August 3, 2023

No. 1591, July 29, 2023

No. 1590, July 24, 2023

No. 1589, July 19, 2023

No. 1588, July 14, 2023

Pure Poem No.1609

II III II II III II II I I III II III I I I I III III II  
I II I I II I I III III II I II III III III III III II II I  
III I III III I III III II II I III I II II II II II I I III  
III I I III I II II I III I II II I II III III I II III II  
II III III II III I I III II III I I III I II II III I II I  
I II II I II III III II I II III III II III I I II III I III  
I I I I I II III III II III II III II I III II III II I  
III III III III III I II II I II I I II I III II I II I III  
II II II II III I I III I III III I III II I III I III II  
I II III III III III I III II II II I I II I II II I III I  
III I II II II III II I I I III III I III I I III II III  
II III I I I I II I III III III II III II III III II I II  
II II I II III I I II I II III I I III III III II III I III  
I I III I II III III I III I II III III I II II I II III II  
III III II III I II II III II III I II II III I I III I II I  
II II III I III II II I III I I I III I II III III II II I  
I I II III II I I III II III III III II III I II II I I III

# Nick Montfort “Concrete Perl”

[nickm.com](#) > [poems](#)

## Concrete Perl

A set of four concrete poems realized as 32-character Perl programs, by Nick Montfort

```
h      dd  kxvdrkypsbabankd uuvrcqiez jssvhtlirkknk  nn  m
      zb  qb kxmduzfs gpuzvy  vmf siuppz rntkf bhvqlxwhxfx ciw  vf
kh lai  oqsznzuncl w  d  adamjbe mnbqouoen s rboj  bqqtqs fnif  ul
```

- [All the Names of God](#)
- [Alphabet Expanding](#)
- [ASCII Hegemony](#)
- [Letterformed Terrain](#)

You can download the linked Perl files and/or simply copy and paste the following four lines, which correspond to the four titles above:

**All the Names of God:**

```
perl -e '{print"a"x++$...$"x$.,$,=,;redo}'
```

**Alphabet Expanding:**

```
perl -e '{print$,$x($,+=.01),a..z;redo}'
```

**ASCII Hegemony:**

```
perl -e '{print" ".chr for 32..126;redo}'
```

**Letterformed Terrain:**

```
perl -e '{print$,$_=(a..z)[rand$=];redo}'
```