

InstallTextのプラグインの使い方

はじめに
このプラグインはM VとM Z 共用です。

プラグインの特徴
このプラグインはRPGツクールでのシナリオ作成を補助するツールです。

プラグインコマンドを用いてテキストファイルを直接メッセージウィンドウに表示する機能のため、デバックのときにゲームを再起動しなくてもテキストの内容が反映されます。
※スクロールウィンドウには対応していません。

また、本文中で2回以上連続改行による自動改ページ、特定の文字列を変数登録する機能によって、直感的で効率的なシナリオ作成を可能とします。

このヘルプではチュートリアル形式で使い方を説明していきます。

プラグインの設定

基本設定

名前:

InstallText

状態:

ON

作者:

こばた ふたる

シナリオをテキストから直接文章に表示するプラグイン。

ヘルプ

プラグインコマンド
phk_installText build ファイル名 : テキストを読み込みプログラムを立ち上げ初
phk_installText init : シナリオ進行を初期化 (テキストの最初からに
phk_installText start : シナリオ進行を開始・stopの後から再開
phk_installText jump ラベル名 : labelに飛ぶ
phk_installText end : シナリオ進行を終了
phk_installText destruct : インスタンスの破壊
phk_installText debug index : 何番目の文章またはタグを参照しているか
phk_installText debug done : シナリオが進行しているかどうか
phk_installText_convert ファイル名: 指定のファイルのいテキスト変数などを展開

【タグとコマンド】

テキストに記述するイベントタグの記法
#e{stop} : シナリオの進行を一時的に抜ける
#l{ラベル名} : ラベル {} の中に任意のラベル名を入れる
#e{end} : シナリオの終了

パラメータ

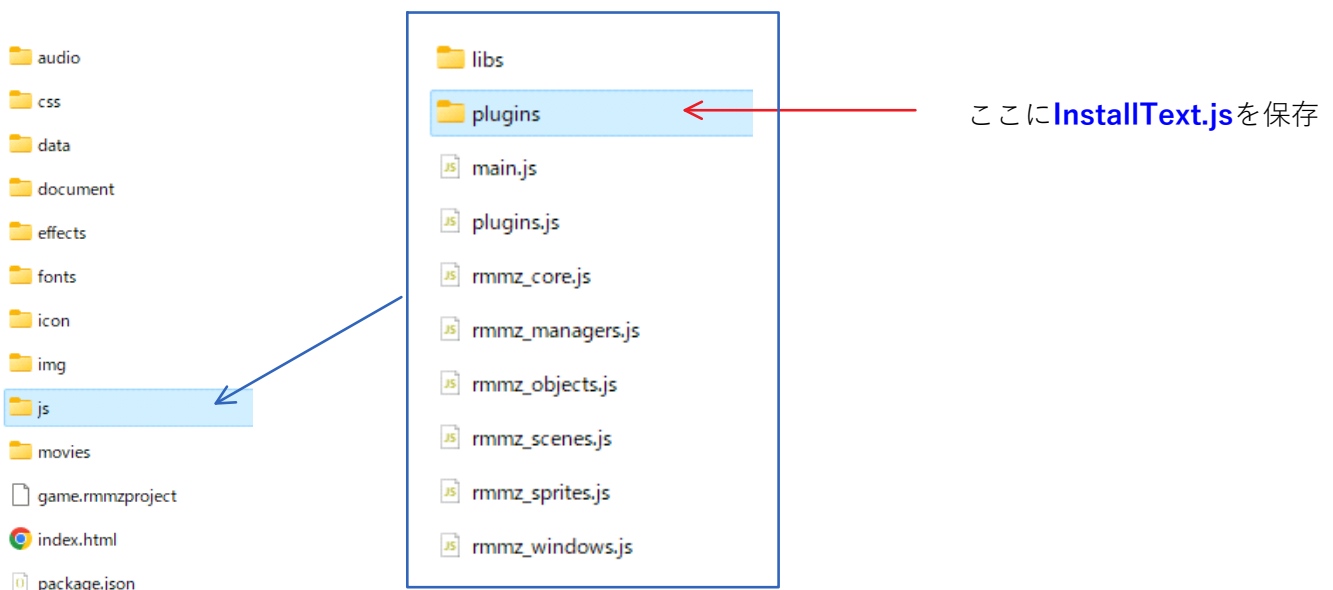
名前	値
参照フォルダ名	document
パラメーター有効化	{ "act_textTag": "true", "act_event...
定義	
テキストタグ	{ "variable": "\$v", "unicode": "\$u", ...
イベントタグ	{ "event": "#e", "label": "#l" }
ウィンドウタグ	{ "windowBack": "!b", "windowPotion...
コメントタグ	{ "commentOut": "///" }
テキストタグ設定	
テキスト変数の設定	[]
色制御文字の設定	[]
unicode文字の設定	[]
一時停止・終了タグの設定	{ "stopScenario": "stop", "endScena...
テキストウィンドウの設定	{ "window_window": "通常", "window_...

OK

キャンセル

プラグインを有効化しよう

一般的なプラグインの有効化の説明なので知ってる人は読み飛ばしてOK



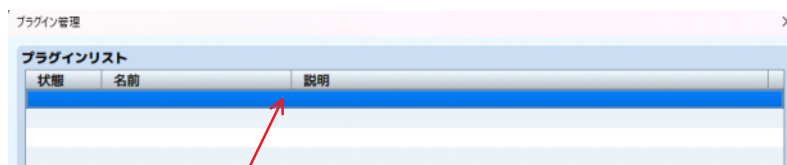
MV



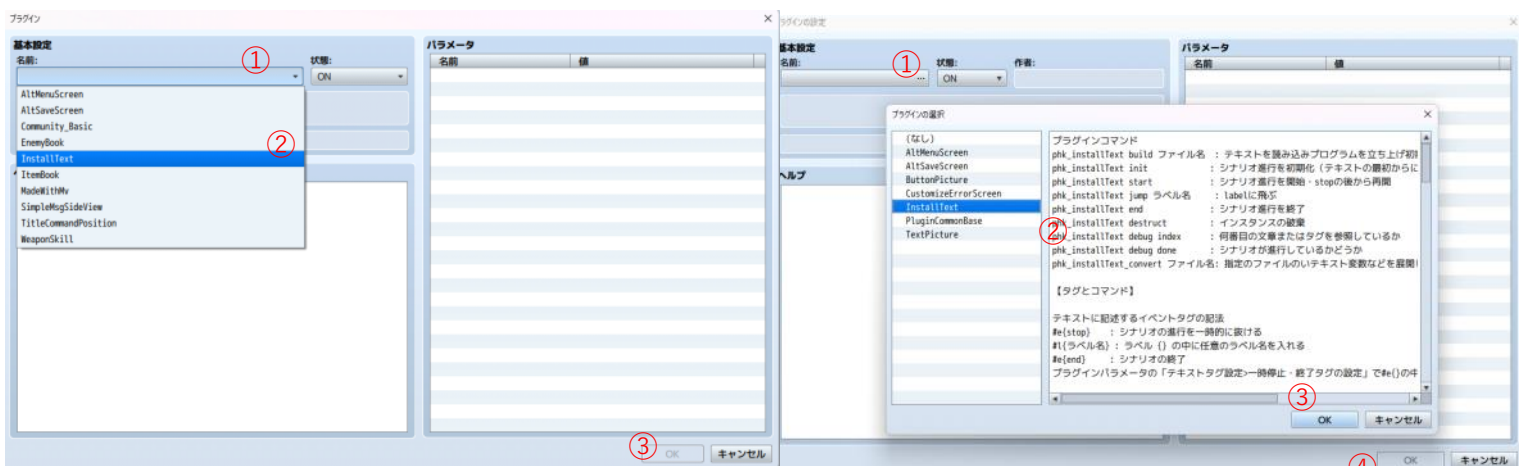
MZ



エディタを起動
このボタンをクリック

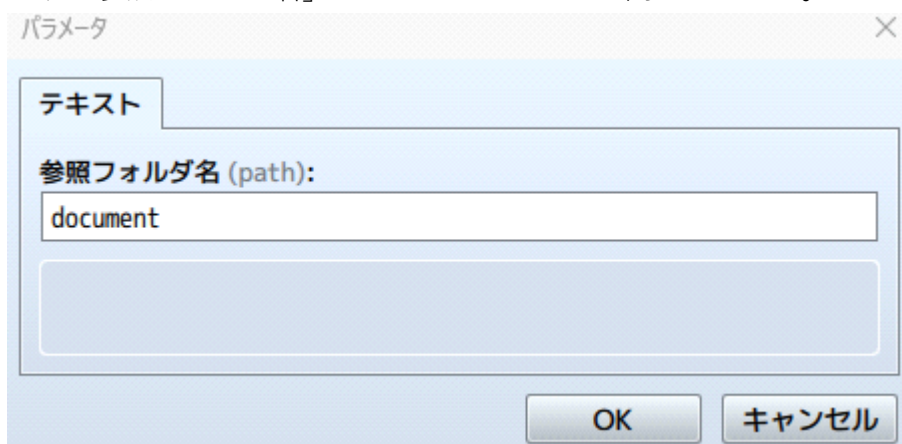


ここをダブルクリック



テキストを保存するフォルダを指定しよう

まず「参照フォルダ名」というパラメーターを開いてみよう。



パラメータ












テキスト

参照フォルダ名 (path):

document

OK キャンセル

デフォルトはdocumentです。デフォルトで使用する場合はdocumentという名前のフォルダをプロジェクトフォルダ内に作成し、そこにテキストファイルを保存してください。


 audio	2022/12/28 16:17	ファイル フォルダ	
 data	2022/12/28 21:48	ファイル フォルダ	
 document	2022/12/29 3:16	ファイル フォルダ	
 fonts	2022/12/28 16:17	ファイル フォルダ	
 icon	2022/12/28 16:17	ファイル フォルダ	
 img	2022/12/28 16:17	ファイル フォルダ	
 js	2022/12/28 16:17	ファイル フォルダ	
 movies	2022/12/28 16:17	ファイル フォルダ	
 Game.rpgproject	2022/12/29 3:05	RPGMV Project	1 KB
 index.html	2022/12/29 3:05	Chrome HTML Do...	2 KB
 package.json	2022/12/27 22:27	JSON ソース ファイル	1 KB

変更する場合は、テキストを保存するフォルダをプロジェクトフォルダ内に作成し、そのフォルダ名をパラメータ（参照フォルダ名）に入力してください。

テキストを書いてみよう

ここからは実際にプラグインコマンドを使ってみます。

以下のテキストをコピペしたexample.txtを作成してdocumentフォルダ（参照フォルダ）に保存してください。文字コードはutf-8で。

名前	更新日時	種類	サイズ
 example.txt	2023/01/14 19:01	テキストドキュメント	1 KB

コピペ用テキスト

主人公

「このプラグインの使い方がわからないのよね」

???

「わかったぜ。今日はこの¥c[1]プラグイン¥c[0]について説明するぜ」

※行間が削除されてしまう場合はすいませんがうまく追加してください

次のページから説明する内容

文章の読み込みをしてみる

改ページのルール

コメントアウト

いったんテキストを抜けてイベントを挟む

選択肢とラベルジャンプ

ウインドウの設定を変更してみる

顔グラをつけてみる

名前欄を使ってみる

変数を設定してみよう



MZでのプラグイン設定画面

文章の読み込みを試みる

イベントに以下のようにプラグインコマンドを設定してください。

MV

- ◆注釈：テキストファイルの読み込み
- ◆プラグインコマンド：phk_installText build example.txt
- ◆注釈：シナリオ進行の開始
- ◆プラグインコマンド：phk_installText start

MZ

- ◆注釈：テキストファイルの読み込み
- ◆プラグインコマンド：InstallText, テキストファイルの読み込み
：：ファイル名 = example.txt
- ◆注釈：シナリオ進行の開始
- ◆プラグインコマンド：InstallText, テキスト進行

保存したらテストプレイを起動してイベントを実行してみてください。

文章について

テキストに書かれた文章はそのまま表示されます。
また制御文字もそのままの表記で使えます。

改ページのルール

文章が2回以上連続で改行れている（つまり行間が1行以上あいていると）と余分な改行は削除されてそこで改ページされます。

あえてそれを回避したい場合は行間にスペース（空白文字）をいれてください。

もちろん5行以上の場合はツクルの仕様にに基づき改ページとなります。

文章 1 ……文章 1 で 1 ページ

文章 2 ……文章 2 で 1 ページ

文章3 ……文章3～4の2行で1ページ
文章4

文章 5 ……文章 5～6 の 3 行で 1 ページ
□ ……スペース
文章 6

コメントアウト

ここからはテキストコマンドを使用していきます。
以下の文章を追記して、イベントを実行してみてください。

```
主人公  
「このプラグインの使い方がわからないのよね」  
  
???  
「わかったぜ。今日はこの%c[1]プラグイン%c[0]について説明するぜ」
```

```
//{コメントアウト}
```

```
???  
「コメントアウトの書き方は上の通りだぜ」
```

//{}で囲われた部分はコメントアウト（注釈）となりメッセージには表示されません。

例えば

文章 1 //{注釈}文章 2

としても//{}は除外されます。

いったんテキストを抜けてイベントを挟む

まずは以下の文章を追記してください。

```
主人公
「このプラグインの使い方がわからないのよね」

???
「わかったぜ。今日はこの%c[1]プラグイン%c[0]について説明するぜ」

//[コメントアウト]

???
「コメントアウトの書き方は上の通りだぜ」

//[ここでジャンプ}
#e{stop}

???
「いったんイベントを抜けるのはこうするんだぜ」
```

次に以下の赤枠をイベントに追加してください。

保存したらテストプレイを再起動(F5)してイベントを実行してください。

MV

```
◆注釈：テキストファイルの読み込み
◆プラグインコマンド：phk_installText build example.txt
◆注釈：シナリオ進行の開始
◆プラグインコマンド：phk_installText start
◆注釈：イベントの挿入
◆移動ルートの設定：このイベント（飛ばす，ウェイト）
：                ：◇ジャンプ：+0, +0
：                ：◇ジャンプ：+0, +0
◆注釈：シナリオ進行の再開
◆プラグインコマンド：phk_installText start
```

MZ

```
◆注釈：テキストファイルの読み込み
◆プラグインコマンド：InstallText, テキストファイルの読み込み
：                ：ファイル名 = example.txt
◆注釈：シナリオ進行の開始
◆プラグインコマンド：InstallText, テキスト進行
◆注釈：イベントの挿入
◆移動ルートの設定：このイベント（飛ばす，ウェイト）
：                ：◇ジャンプ：+0, +0
：                ：◇ジャンプ：+0, +0
◆注釈：シナリオ進行の再開
◆プラグインコマンド：InstallText, テキスト進行
```

文中の**#e{stop}**でテキストをいったん抜けて、プラグインコマンド**start/テキスト進行**で途中から再開できます。

その間に文章以外のアクションを挟むことができます。

選択肢とラベルジャンプ

まずは以下の文章を追記してください。

<pre>主人公 「このプラグインの使い方がわからないのよね」 ??? 「わかったぜ。今日はこの%c[1]プラグイン%c[0]について説明するぜ」 //{コメントアウト} ??? 「コメントアウトの書き方は上の通りだぜ」 //{ここでジャンプ} #e{stop} ??? 「いったんイベントを抜けるのはこうするんだぜ」 //{長くなったので横に移動しました}</pre>	<pre>??? 「わかったか？」 #e{stop} //{選択肢 #l←これはエルです} #l{はい} ??? 「よかったぜ」 #e{end} #l{いいえ} ??? 「わかれ」</pre>
---	--

次に以下の赤枠をイベントに追加してください。
保存したらテストプレイを再起動(F5)してイベントを実行してください。

◆注釈：テキストファイルの読み込み
◆プラグインコマンド：phk_installText build example.txt
◆注釈：シナリオ進行の開始
◆プラグインコマンド：phk_installText start
◆注釈：イベントの挿入
◆移動ルートの設定：このイベント（飛ばす、ウェイト）
：◇ジャンプ：+0, +0
：◇ジャンプ：+0, +0
◆注釈：シナリオ進行の再開
◆プラグインコマンド：phk_installText start
◆注釈：選択肢の表示
◆選択肢の表示：はい、いいえ（ウィンドウ、右、#1, #2）
：はいのとき
◆注釈：#l{はい}にラベルジャンプ
◆プラグインコマンド：phk_installText jump はい
◆
：いいえのとき
◆注釈：#l{いいえ}にラベルジャンプ
◆プラグインコマンド：phk_installText jump いいえ
◆
：分岐終了

◆注釈：テキストファイルの読み込み
◆プラグインコマンド：InstallText, テキストファイルの読み込み
：ファイル名 = example.txt
◆注釈：シナリオ進行の開始
◆プラグインコマンド：InstallText, テキスト進行
◆注釈：イベントの挿入
◆移動ルートの設定：このイベント（飛ばす、ウェイト）
：◇ジャンプ：+0, +0
：◇ジャンプ：+0, +0
◆注釈：シナリオ進行の再開
◆プラグインコマンド：InstallText, テキスト進行
◆注釈：選択肢の表示
◆選択肢の表示：はい、いいえ（ウィンドウ、右、#1, #2）
□：はいのとき
◆注釈：#l{はい}にラベルジャンプ
◆プラグインコマンド：InstallText, ラベルジャンプ
：ラベル名 = はい
◆
□：いいえのとき
◆注釈：#l{いいえ}にラベルジャンプ
◆プラグインコマンド：InstallText, ラベルジャンプ
：ラベル名 = いいえ
◆
：分岐終了

#e{stop}でいったん抜けた後、選択肢の表示を行い、それぞれプラグインコマンドjump/ラベルジャンプで指定の#l{ラベル}に飛べます。また、途中でシナリオを終了するには#e{end}を使用します。

ウィンドウの設定を変更してみる

まずは新しくexample2.txtを作成して以下をコピーして保存してください。
文字コードはutf-8で。

```
!b{暗く}  
!p{中}  
¥{プラグイン講座 ¥}～第二幕～  
  
¥c[2]～中級編～¥c[0]  
!b{通常}  
!p{下}  
  
???  
「ということだぜ」
```

次に以下のイベントをマップに作成してください。
そしたら保存してテストプレイでイベントを実行してみてください。

MV

◆注釈：テキストファイルの読み込み

◆プラグインコマンド：phk_installText build example2.txt

◆注釈：シナリオ進行の開始

◆プラグインコマンド：phk_installText start

◆

◆注釈：テキストファイルの読み込み

◆プラグインコマンド：InstallText, テキストファイルの読み込み

： ファイル名 = example2.txt

◆注釈：シナリオ進行の開始

◆プラグインコマンド：InstallText, テキスト進行

!b{通常/暗く/透明}でバックグラウンドを、**!p{上/中/下}**でウィンドウの位置を変更できます。

設定を行った次の行から有効になります。

また、新たに設定し直すまでこの設定はずっと有効になります。

顔グラをつけてみる

まずは以下の文章を追記してイベントを実行してください。

```
!b{暗く}  
!p{中}  
¥{プラグイン講座 ¥}～第二幕～
```

```
¥c[2]～中級編～¥c[0]  
!b{通常}  
!p{下}
```

```
???  
「ということだぜ」
```

```
!f{Actor1, 2}
```

```
???  
「顔グラは表示されたか？  
画像ファイル名と番号で指定できるぜ」
```

```
???  
「それじゃあ解除するぜ」
```

```
!f{,0}
```

```
???  
「これで消えたかな」
```

!f{画像ファイル名, 画像番号}でウィンドウに顔グラをつけることができます。

顔グラを消すには**!f{,0}**としてください。（コンマを忘れずに！）

こちらも新たに設定するまではずっと有効になります。

名前欄を使ってみる

まずは以下の文章を追記してイベントを実行してください。

<p>!b{暗く} !p{中} ¥(プラグイン講座 ¥)～第二幕～</p> <p>¥c[2]～中級編～¥c[0] !b{通常} !p{下}</p> <p>??? 「ということだぜ」</p> <p>If{Actor1, 2} ??? 「顔グラは表示されたか？」 画像ファイル名と番号で指定できるぜ」</p> <p>??? 「それじゃあ解除するぜ」</p> <p>If{,0} ??? 「これで消えたかな」</p>	<p>!s{¥c[2] ???} 「つぎに名前欄の使用方法だぜ」</p> <p>「名前欄はMZにしかない機能だがコマンドはMVでも有効なんだぜ」</p> <p>「MVの場合は本文の先頭に自動的に追記されるぜ」</p> <p>「テキストのセリフの前に¥c[2]『 ??? 』 ¥c[0]はないのに表示されてる文章にはあるのに気づいたか？」</p> <p>!s{}</p> <p>主人公 「た、たしかに！」</p>
---	---

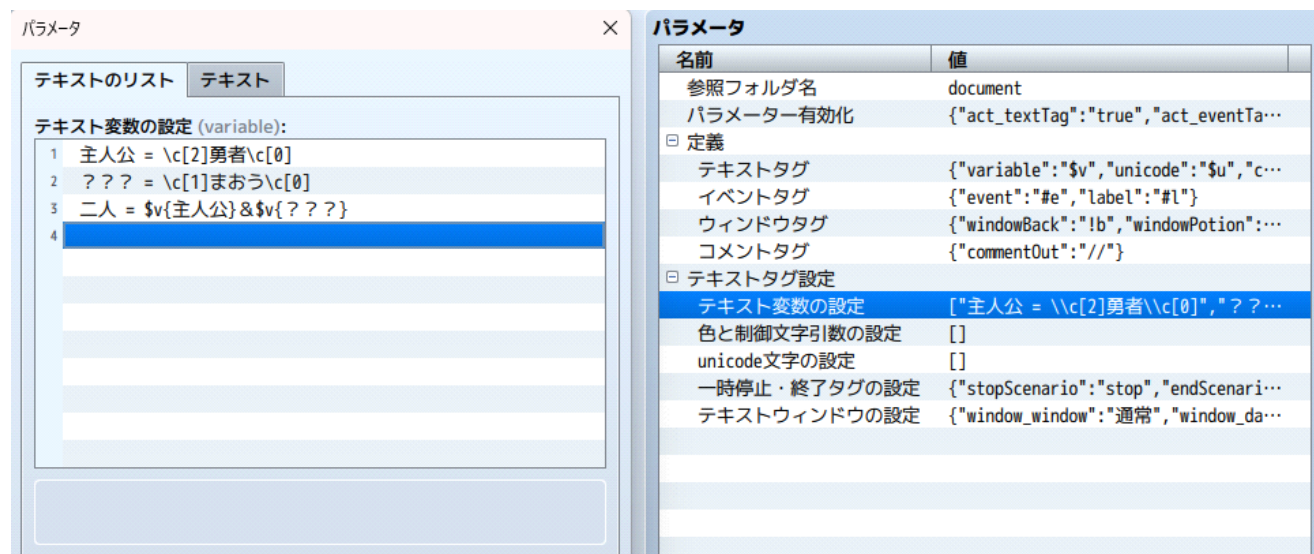
!s{名前}とすることでMZでは**名前欄に「名前」を表示**することができます。
一方でMVでは**文頭に自動的に名前が追加**されます。

名前欄を消すには**!s{}**としてください。

こちらも新たに設定するまではずっと有効になります。
そのため同じ人物が喋っているシーンでは記述量がぐっと減ります。

変数を設定してみよう

まずプラグイン>パラメータ>テキスト変数の設定を以下のように記述してください。
イコールは半角です。イコール前後のスペースはあってもなくてもいいです。



新しく example2.txt を作成して以下をコピペして保存してください。文字コードは utf-8 で。

???

「ついに最後の試練、変数だ。最後なので私の正体を明かすが
実は、ま……\$v{???}なんだ」

主人公

「\$v{主人公}たる私は\$v{???}に教わっていたのか。
だたこれで使い方もマスターできたし」

\$v{二人}

\$c{紫}「そろそろ狩るか\$u{スペード黒}」

次に以下のイベントをマップに作成してください。
そしたら保存してテストプレイでイベントを実行してみてください。

MV

- ◆注釈：テキストファイルの読み込み
- ◆プラグインコマンド：phk_installText build example3.txt
- ◆注釈：シナリオ進行の開始
- ◆プラグインコマンド：phk_installText start
- ◆

MZ

- ◆注釈：テキストファイルの読み込み
- ◆プラグインコマンド：InstallText, テキストファイルの読み込み
： ファイル名 = example3.txt
- ◆注釈：シナリオ進行の開始
- ◆プラグインコマンド：InstallText, テキスト進行

パラメータ>テキスト変数の設定で設定した**変数**はテキストで**\$v{変数名}**で表記できます。
またデフォルトで**\$c{青}**などで**色制御文字**を、**\$u{変数名}**で文字化けしやすい**unicode**を表示でき
この二つはデフォルトで設定されています。それぞれの設定で変数の追加や変更ができます。

色制御文字\$¥c{}一覧表

¥c[0]	white	白
¥c[1]	lightblue	薄青
¥c[2]	lightred	薄赤
¥c[3]	lightgreen	薄緑
¥c[4]	lightcyan	ライトシアン
¥c[5]	lightpurple	薄紫
¥c[6]	lightyellow	浅黄
¥c[7]	gray	灰
¥c[8]	silver	銀
¥c[9]	blue	青
¥c[10]	red	赤
¥c[11]	green	緑
¥c[12]	sky	空
¥c[13]	purple	紫
¥c[14]	yellow	黄
¥c[15]	black	黒

¥c[16]	aqua	水
¥c[17]	lemon	レモン
¥c[18]	scarlet	紅
¥c[19]	navy	紺
¥c[20]	orange	オレンジ
¥c[21]	gold	金
¥c[22]	cobalt	コバルト
¥c[23]	cyan	シアン
¥c[24]	lightlime	ライトライム
¥c[25]	brown	茶
¥c[26]	wistaria	藤
¥c[27]	pink	ピンク
¥c[28]	darkgreen	濃緑
¥c[29]	lime	ライム
¥c[30]	darkpurple	濃紫
¥c[31]	violet	堇



unicode文字\$¥u{}一覧表

¥u2660	black spade	スペード黒	spade	スペード
¥u2663	black club	クローバー黒	club	クローバー
¥u2665	black heart	ハート黒	heart	ハート
¥u2666	black diamond	ダイヤ黒	diamond	ダイヤ
¥u2664	white spade	スペード白		
¥u2667	white club	クローバー白		
¥u2661	white heart	ハート白		
¥u25C7	white diamond	ダイヤ白		

テキストコマンド一覧

コマンド	記法	引数 ({}の中)	説明
注釈	//{}	[任意の文字列]	この部分は表示されない
イベント	#e{}	stop/end	文章の進行を一時停止/終了
ラベルジャンプ	#l{}	[ラベル名]	ジャンプ先のラベル
選択肢の表示	#o{} 	[選択肢1 (=>ラベル1)] [選択肢2 (=>ラベル2)]	, /改行で区切った選択肢を表示 末尾の分岐先の表記は省略可能
ウインドウ位置	!p{} 	上/中/下	ウインドウ位置を設定
ウインドウ背景	!b{} 	通常/暗く/透明	ウインドウ背景を設定
ウインドウの顔グラ	!f{} 	[画像ファイル名], [画像番号]	ウインドウの顔グラを設定
ウインドウの名前欄	!s{} 	[任意の文字列]	ウインドウ名前欄を設定
ユーザー定義テキスト変数	\$v{} 	[任意の文字列]	変数の表示
色制御文字テキスト変数	\$c{} 	[色の名前]	変数の表示
unicodeテキスト変数	\$u{} 	[任意の文字列]	変数の表示

プラグインコマンド一覧

MV	MZ	引数	説明
phk_installText build	テキストファイルの読み込み	[ファイル名]	ファイル読み込み 初期化も同時に行う
phk_installText init	テキスト進行初期化	-	テキストの進行を最初からにする。
phk_installText start	テキスト進行	-	テキスト進行を開始する。 または#e{stop}の続きから再開する。
phk_installText end	テキスト終了	-	テキスト進行を終了する。
phk_installText jump	ラベルジャンプ	[ラベル名]	ラベルの行にジャンプする。
phk_installText destruct	テキストデータの解放	-	読み込んだテキストデータを イベントから削除する。
phk_installText debug	デバッグ	index/done	index：何番目のテキストを読んでるか done：テキスト進行中かどうか
phk_installText convert	変換テキスト出力	[ファイル名]	文章の表示の一括入力用のテキストを 生成、出力する。