

dvipdfmx

と

3っ

の

バッド・ノウハウ

八登 崇之 (Takayuki YATO)

Twitter ID: [@zr_tex8r](#)

TeX ユーザの集い 2014

2014年11月8日

この物語 (LT) は

dvipdfmxで

画像を挿入するときの

“封印”された

バッドノウハウを

記したものである。

バッドノウハウ?

バッドノウハウ

||

バグ技

正攻法

VS

バッド・ノウハウ



正攻法

VS

バッド・ノウハウ

```
\begin{document}
```

正しいコード

```
\end{document}
```

“仕様に従った”

正攻法

VS

バッド・ノウハウ

```
\begin{document}
```

正しいコード

```
\end{document}
```



正しい出力結果

正攻法

VS

バッド・ノウハウ

```
\begin{document}
```

正しいコード

```
\end{document}
```



正しい出力結果



当然。

正攻法

VS

バッド・ノウハウ

```
\begin{document}
```

正しいコード

```
\end{document}
```



正しい出力結果



当然。

```
\begin{document}
```

マチガッテルコード

```
\end{document}
```

“仕様は未定義”

正攻法

VS

バッド・ノウハウ

```
\begin{document}
```

正しいコード

```
\end{document}
```



正しい出力結果

```
\begin{document}
```

マチガッテルコード

```
\end{document}
```



正しい出力結果



当然。

正攻法

VS

バッド・ノウハウ

```
\begin{document}
```

正しいコード

```
\end{document}
```



正しい出力結果



当然。

```
\begin{document}
```

マチガッテルコード

```
\end{document}
```

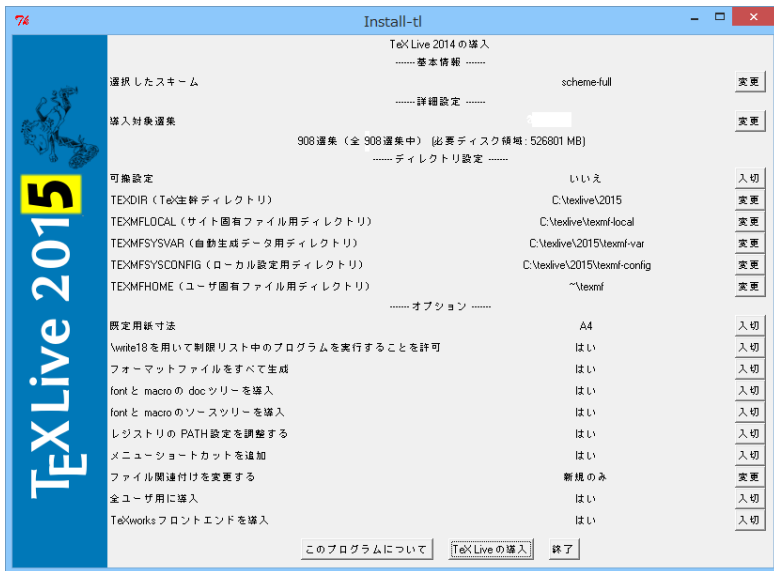


正しい出力結果



素敵。

バージョンアップ!



正攻法

VS

バッド・ノウハウ

```
\begin{document}
```

正しいコード

```
\end{document}
```

```
\begin{document}
```

マチガッテルコード

```
\end{document}
```

“仕様に従った”

“仕様は未定義”

正攻法

VS

バッド・ノウハウ

```
\begin{document}
```

正しいコード

```
\end{document}
```



正しい出力結果



当然。

```
\begin{document}
```

マチガッテルコード

```
\end{document}
```

“仕様は**未定義**”

正攻法

VS

バッド・ノウハウ

```
\begin{document}
```

正しいコード

```
\end{document}
```



正しい出力結果



当然。

```
\begin{document}
```

マチガッテルコード

```
\end{document}
```



間違っ**た**出力結果



破滅!

封印された
バッド・ノウハウは

破滅

を生む



バッド・ノウハウ その①

ドライバ詐称

false driver name

ドライバ指定 ?

ドライバ指定

```
\documentclass[uplatex]{jsarticle}  
\usepackage[dvipdfmx]{graphicx}  
\usepackage{lisp-on-tex}  
%...(以下略)...
```



コレ!

普通のドライバ指定

```
\documentclass[uplatex]{jsarticle}  
\usepackage[dvipdfmx]{graphicx}  
\usepackage{lisp-on-tex}  
%...(以下略)...
```



一致!

```
[zr@here]$ dvipdfmx talk.dvi
```

バッドなドライバ指定

```
\documentclass[uplatex]{jsarticle}  
\usepackage[dvips]{graphicx}  
\usepackage{lisp-on-tex}  
%...(以下略)...
```



食い違う!!

```
[zr@here]$ dvipdfmx talk.dvi
```

バッドなドライバ指定

```
\documentclass[uplatex]{jsarticle}  
\usepackage[dvipdfm]{graphicx}  
\usepackage{lisp-on-tex}  
%...(以下略)...
```



食い違う!!

```
[zr@here]$ dvipdfmx talk.dvi
```

バッドなドライバ指定

```
\documentclass[uplatex]{jsarticle}  
\usepackage[dviout]{graphicx}  
\usepackage{lisp-on-tex}  
%...(以下略)...
```



食い違う!!

```
[zr@here]$ dvipdfmx talk.dvi
```

“ドライバ詐称”してみよう!

```
% upLaTeX 文書
\documentclass[a6paper,papersize,uplatex]{jsarticle}
\usepackage[dvips]{graphicx}
\setlength{\fboxrule}{8pt}
\setlength{\fboxsep}{0pt}
\begin{document}

\begin{center}\LARGE
{\TeX} ユーザの集い 2014
\par\smallskip
\fbox{\includegraphics[width=3cm]{image.eps}}
\par\smallskip
は\\{\gtfamily アレ。}
\end{center}

\end{document}
```


“ドライバ詐称”してみよう!

- ▶ ドライバ指定は“dvips”

% upLaTeX 文書

```
\usepackage[dvips]{dvipdfmx}
```

\setlength{\fboxsep}{0pt}

- ▶ EPS 画像を挿入する

```
\includegraphics[width=3cm]{image.eps}
```

\fbox{\includegraphics[width=3cm]{image.eps}}

\par\smallskip

は\{\gtfamily アレ。

\end{center}

\end{document}



古い $\text{T}_\text{E}\text{X}$ 環境で処理すると…

($\text{T}_\text{E}\text{X}$ Live 2012)

$\text{T}_\text{E}\text{X}$ ユーザの集い 2014



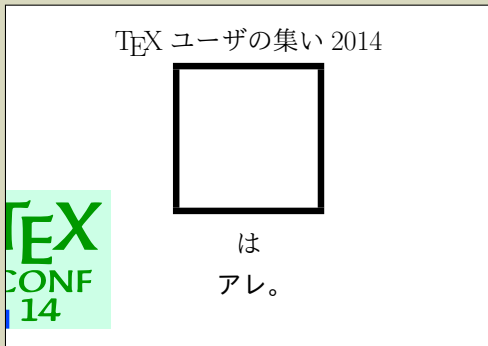
は
アレ。



正常。

新しい $\text{T}_\text{E}\text{X}$ 環境で処理すると…

($\text{T}_\text{E}\text{X}$ Live 2014)



破滅。

楽しい!!

👉('ω'👉)≡👉('ω')👉≡(👉'ω')👉

バッド・ノウハウ その②

バウンディング ボックス詐欺

false bounding box

バウンディングボックス
(bbox) って何？

バウンディングボックス
(bbox) って何

説明省略。

そう、アレです。

各々の画像は固有の
bbox の値を持つ。

image.eps



各々の画像は**固有の**
bbox の値を持つ。

image.eps



- ▶ EPS 形式
- ▶ bbox =
[540 315 900 675]

各々の画像は固有の
bbox の値を持つ。

image.png



- ▶ PNG 形式
- ▶ bbox =
[0 0 360 360]

ビットマップ画像も同様。

普通のbbox 指定の方法。

image.png



▶ PNG 形式

▶ bbox =

[0 0 360 360]



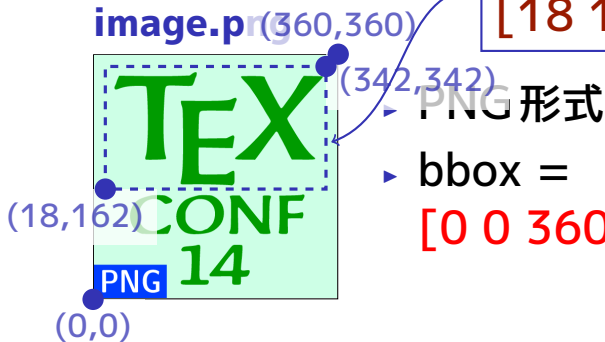
普通。

```
\includegraphics[bb=0 0 360 360]{image.png}
```

画像を“一部だけ”挿入したい!

“一部” =

[18 162 342 342]



▶ PNG 形式
▶ bbox =
[0 0 360 360]

バッドなbbox指定の方法。

image.png



“一部” =

[18 162 342 342]

▶ PNG形式

▶ bbox =

[0 0 360 360]



食い違う!

```
\includegraphics[bb=18 162 342 342]{image.png}
```

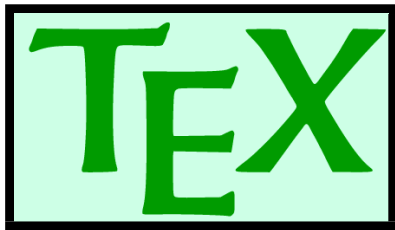
“bbox 詐称”してみよう!

- ▶ ドライバは当然“dvipdfmx”。
- ▶ さっきの例の通りに
PNG 画像を“一部だけ”挿入。

```
\includegraphics  
[bb=18 162 342 342]{image.png}
```

古いT_EX環境で処理すると…

(T_EX Live 2012)



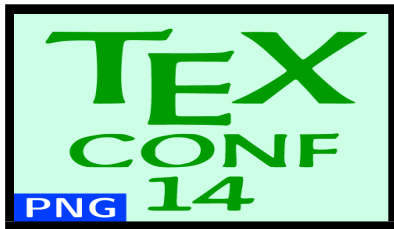
は
アレ。



正常。

新しい $\text{T}_\text{E}\text{X}$ 環境で処理すると…

($\text{T}_\text{E}\text{X}$ Live 2014)



は
アレ。



破滅。

楽しい!!!

✌('ω'✌)≡✌('ω')✌≡(✌'ω')✌

バッド・ノウハウ その③

面倒なので ポイント単位

lazily-in-point

大昔の $\text{T}_\text{E}\text{X}$ 環境では…

(2005 年頃の $\text{T}_\text{E}\text{X}$ 環境)



アレ。



正常。

新しいT_EX環境では…

(T_EX Live 2014)



アレ。



正常!

少し古い $\text{T}_\text{E}\text{X}$ 環境では…

($\text{T}_\text{E}\text{X}$ Live 2012)



アレ。



破滅。

楽しい…？

👉('ω'👉)≡👉('ω')👉≡(👉'ω')👉

楽しくない!!!



正しいコードを書こう！



まさに
正論。

ステップ ②

dvipdfmx
を使う

dvipdfm

~~dvipdfm~~

dvipdfmx

ステップ①

extractbb
自動起動設定

ステップ②

graphicx
読込

graphicx (≡ **dvipdfmx**。)

```
\usepackage[dvipdfmx]{graphicx}
```

color にも **dvipdfmx**。

```
\usepackage[dvipdfmx]{graphicx}
```

```
\usepackage[dvipdfmx]{color}
```

ステップ ③

画像を挿入

bb は無し。

```
\includegraphics{image.png}
```

“一部だけ” → **viewport**。

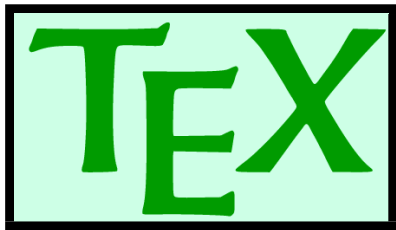
```
\includegraphics
```

```
[viewport=18 162 342 342]
```

```
\{image.png}
```



大成功。



TEX

は
アレ。





Happy
TeXing!

