

**position: fixed;**

**チヨツトデキル**



**Back to Basics**



伊藤 由暁  
(越智)

株式会社まぼろし

@o\_ti

<http://dskd.jp>

**position: fixed;**

チョットデキルために  
知っておきたい5つのこと

- fixedの仕様
- ブラウザの実装
- ブラウザのバグ
- fixedあるある
- 対策

**position: fixed;の仕様**

**CSS2.1では**  
**「Visual Formatting Model」 内で**  
**定義されている**

# Positionプロパティにおける fixed値の定義 (CSS2.1)

## 9 Visual formatting model - 9.3.1 Choosing a positioning scheme: 'position' property

<http://www.w3.org/TR/CSS2/visuren.html#choose-position>

### “ fixed

The box's position is calculated according to the 'absolute' model, but in addition, the box is fixed with respect to some reference. As with the 'absolute' model, the box's margins do not collapse with any other margins. In the case of handheld, projection, screen, tty, and tv media types, the box is fixed with respect to the viewport and does not move when scrolled. In the case of the print media type, the box is rendered on every page, and is fixed with respect to the page box, even if the page is seen through a viewport (in the case of a print-preview, for example). For other media types, the presentation is undefined. Authors may wish to specify 'fixed' in a media-dependent way. For instance, an author may want a box to remain at the top of the viewport on the screen, but not at the top of each printed page. The two specifications may be separated by using an @media rule, as in:

```
@media screen {  
  h1#first { position: fixed }  
}  
@media print {  
  h1#first { position: static }  
}
```

UAs must not paginate the content of fixed boxes. Note that UAs may print invisible content in other ways. See "Content outside the page box" in chapter 13.





日本語で読もう。

## 【邦訳】 9 視覚整形モデル - 9.3.1 位置決めスキームの選択：'position'プロパティ

<http://momdo.github.io/css2/visuren.html#choose-position>

### “ fixed

ボックスの位置は'絶対'モデルに従って計算されるが、それに加えて、ボックスはある参照に対して固定される。'絶対'モデルの場合と同じように、ボックスのマージンは他のマージンと相殺しない。

handheld、projection、screen、tty、tvメディアタイプの場合、ボックスはビューポートに対して固定され、スクロール時に移動しない。



## 【邦訳】 9 視覚整形モデル - 9.3.1 位置決めスキームの選択：'position'プロパティ

<http://momdo.github.io/css2/visuren.html#choose-position>

“ printメディアタイプの場合、ボックスは各ページごとにレンダリングされ、たとえページがビューポートを通して見られる場合でもページボックスに対して固定される（たとえば、印刷プレビューの場合）。他のメディアタイプの場合、見栄えは未定義である。著者はメディアごとに'fixed'を指定したい場合もあるだろう。たとえば、著者が画面上のビューポートの上辺に対してボックスを固定したいが、印刷ページでは固定したくない場合である。2つの仕様は@media規則を用いて分離されてもよい：

```
@media screen {  
  h1 #first { position: fixed }  
}  
@media print {  
  h1 #first { position: static }  
}
```

ユーザーエージェントは、固定ボックスのコンテンツを割り付けてはならない。ユーザーエージェントは別の方法で不可視のコンテンツを印刷してもよいことに注意する。13章の"ページボックスの外側のコンテンツ"を参照のこと。

## 補足：なぜCSS2.2の邦訳を参照しているのかについて

今回の発表では、W3CのオリジナルではCSS2.1を参照し、邦訳はCSS2.2のものを参照しました。これは、邦訳のCSS2.1には一部に翻訳にミスがあるのが直接の理由です。このことについて訳者のもんどさん

(@momdo\_) に伺ったところ、**「2.2訳は全文一通り訳者が見直した（訳が改善されているところがある）ので、2.1訳より2.2訳を勧めます」**と回答をいただいています。訳者の言うように、**CSS2.1の邦訳でもんどさんの文書**を参照する場合は、**もんどさんのCSS2.2の邦訳**を見るようにしましょう。CSS2.1訳とCSS2.2訳の差分はこちらに書かれているので、参考にするときの助けにしてください。 <http://momdo.github.io/css2/changes.html>

そして、日本語訳は参考情報であることに十分注意してください。オリジナルはあくまでW3Cの英文です。

※このページは発表後にフォローアップとして追加しました

# 固定配置の定義 (CSS2.1)

## 9 Visual formatting model - 9.6.1 Fixed positioning

<http://www.w3.org/TR/CSS2/visuren.html#fixed-positioning>

“ Fixed positioning is a subcategory of absolute positioning. The only difference is that for a fixed positioned box, the containing block is established by the viewport. For continuous media, fixed boxes do not move when the document is scrolled. In this respect, they are similar to fixed background images. For paged media, boxes with fixed positions are repeated on every page. This is useful for placing, for instance, a signature at the bottom of each page. Boxes with fixed position that are larger than the page area are clipped. Parts of the fixed position box that are not visible in the initial containing block will not print.

日本語で。



## 【邦訳】 9 視覚整形モデル - 9.6.1 固定配置

<http://momdo.github.io/css2/visuren.html#fixed-positioning>

“ 固定配置は、絶対配置の下位区分である。固定配置ボックスにとっての唯一の相違点は、包含ブロックがビューポートによって設置されることである。連続メディアに対して、固定ボックスは文書のスクロール時に移動しない。この点において、固定配置ボックスは固定背景画像に似ている。ページメディアに対して、固定配置ボックスはページごとに繰り返される。これは、たとえば各ページの下に署名を打ち込むのに有用である。固定配置ボックスはページの切り取られた領域よりも大きい。初期包含ブロックにおいて不可視である固定配置の一部は、印刷されない。

# **position: fixed;と その他の視覚整形モデルとの関係 (CSS2.1)**

**displayプロパティ**や**floatプロパティ**を持つ要素に  
**positionプロパティ**で**static/relative**値以外を指定した時  
**displayプロパティ**や**floatプロパティ**の値が変化する

**ちゃんと仕様で定義されています！**

# 9 Visual formatting model - 9.7 Relationships between 'display', 'position', and 'float'

<http://www.w3.org/TR/CSS2/visuren.html#dis-pos-flo>



The three properties that affect box generation and layout – 'display', 'position', and 'float' – interact as follows:

1. If 'display' has the value 'none', then 'position' and 'float' do not apply. In this case, the element generates no box.
2. Otherwise, if 'position' has the value 'absolute' or 'fixed', the box is absolutely positioned, the computed value of 'float' is 'none', and display is set according to the table below. The position of the box will be determined by the 'top', 'right', 'bottom' and 'left' properties and the box's containing block.
3. Otherwise, if 'float' has a value other than 'none', the box is floated and 'display' is set according to the table below.
4. Otherwise, if the element is the root element, 'display' is set according to the table below, except that it is undefined in CSS 2.1 whether a specified value of 'list-item' becomes a computed value of 'block' or 'list-item'.
5. Otherwise, the remaining 'display' property values apply as specified.

Specified value	Computed value
inline-table	table
inline, table-row-group, table-column, table-column-group, table-header-group, table-footer-group, table-row, table-cell, table-caption, inline-block	block
others	same as specified

## 【邦訳】 9 視覚整形モデル - 9.7 'display'、'position'、'float'の関係

<http://momdo.github.io/css2/visuren.html#dis-pos-flo>

“

ボックス生成およびレイアウトに作用する3つのプロパティ—'display'、'position'、'float'—は次のように相互に作用する：

1. 'display'が値'none'を持つ場合、'position'および'float'は適用しない。この場合、要素はボックスを生成しない。
2. そうでなければ、'position'が値'absolute'または'fixed'を持つ場合、ボックスは絶対位置であり、'float'の算出値は'none'となり、かつ表示は下記のテーブルに従って設定される。ボックスの位置は、'top'、'right'、'bottom'、'left'プロパティ—およびボックスの包含ブロックによって決定される。
3. そうでなければ、'float'が'none'以外の値を持つ場合、ボックスは浮動または'display'が下記のテーブルに従って設定される。
4. そうでなければ、要素がルート要素である場合、'list-item'の指定値が'block'または'list-item'の算出値になるかどうかCSS 2.2で未定義である場合を除いて、'display'が下記のテーブルに従って設定される。
5. そうでなければ、残りの'display'プロパティ—値は指定された通りに適用される。

その他の資格整形モデルとの関係で  
**position: fixed;**にフォーカスすると

- fixed値が指定されると、floatはnoneとなる
- fixed値が指定されると、displayがinline, inline-block, table-cellなどはdisplay: blockとして扱われる
  - a, span, i要素などもheightやmarginを持てるようになる！
  - display: table-cellでお手軽に上下中央配置をしている要素をfixedにするとvertical-alignの指定が無効になる（要素がdisplay: blockになるから）
- display: tableな要素はfixedしてもtableのまま



**fixed**値を指定すると、デフォルトがインラインレベルの要素でも高さを持つことができる！

指定値	算出値
inline-table	table
<b>inline</b> , table-row-group, table-column, table-column-group, table-header-group, table-footer-group, table-row, <b>table-cell</b> , table-caption, <b>inline-block</b>	block
others	same as specified

仕様を整理

- 座標はabsoluteと同じように計算される
- ある参照に対して固定される→メディアタイプによって参照先が違
  - screenメディアではブラウザ表示領域（viewport）を起点にする
    - スクロール時に移動しない→画面のスクロールに追従しているように見える
  - printメディアではページが複数にまたぐと、ページ毎に所定の位置に印刷される
    - fixedされた要素はページに分断されて印刷されることはない
- displayの値が変更されるものがある（inline→block、table-cell→blockなど）
  - inlineレベルの要素でも高さやマージンを持つことができる
- マージンは相殺しない

ウェブサイトにおいてはviewportに対して絶対配置できるので、コンテンツ量に左右されないレイアウトが可能になる。固定ヘッダーや固定フッター、ページ上部へ戻るリンクの配置に使われることも多い。

# ブラウザの実装

実は10年以上前から使える **position: fixed;**

- **Internet Explorer 7** (2006/10/18 ・ 日本語版は2006/11/2)
- **Firefox 1.0** (2004/12/10)
- **Safari 1.0** (2003/6/24)
- **Opera 4.0** (2000/6/28)
- **Google Chrome 1.0** (2008/12/8)
- **iOS 5.0 Mobile Safari** (2010/11/24)
- **Android 2.2 標準ブラウザ** (2010/5/20)

カッコ内はブラウザのリリース日

※ただし

IE7以降でも、互換モードだとfixedは無効

Androidは2.2、2.3ともに  
metaタグのviewportでuser-scalable=noの指定が必要



まあまあ戦える気がする～✌('ω')✌

しかし現実には甘くない

もちろんAndroidでバグ++

# Android 2.xはfixedチョットデキナイ

- そもそもuser-scalable=noが必要って、ユーザビリティ的に問題がある
- user-scalable=noとposition: fixed;を併用すると、-webkit-transformが効かなくなる※1
- fixedした要素にタッチイベントが発火せず、要素の下にaやinputがあるとそっちに反応してしまう※2

etc...

※1 <http://jsdo.it/esperia/cbhl>

※2 <http://loosely.net/archives/82>

# Android 4.xもfixedチョットデキナイ

- user-scalable=noが不要になったけど、HTMLとCSSの状況によっては必要になる※1
- fixedした要素をdisplay: none;しておき、スクロール後にdisplay: block;に変えると表示位置がおかしい※2
- スクロール中はfixedじゃなくなる端末もある(Galaxy Nexus 4.0.4)

etc...

※1 <http://blog.webcreativepark.net/2013/06/06-112601.html>

※2 <http://lealog.hateblo.jp/entry/2014/12/16/232957>

探せばもっとたくさんのバグが  
見つかることでしょう

<https://www.google.co.jp/search?q=Android+fixed+おかしい>

実はiOSもなんです

- スクロール中にfixedしている要素のz-indexが無視されるバグ (iOS 6, iOS 7) ※1
  - -webkit-transform: translate3d(0,0,0)で治るけど、Androidで別のバグを生む可能性がある
- ページ遷移前にすぐ消える (iOS 6, iOS 7) ※2
- fixedした要素内にinput、textarea、select要素があると、そこにfocusした時にページがスクロールすることがある (iOS 8) ※3
  - focusが発火すると親のfixedした要素がabsoluteにかわるのが原因
- 擬似要素をfixedにすると、スクロール中はfixedにならない (iOS 8) ※4
  - スクロールが終わると指定位置に急に現れる。iOS 8.4.1でも健在

※1 <http://qiita.com/Hieroglyph/items/743781f0b87f66d071c5>

※2 <http://lealog.hateblo.jp/entry/2014/12/16/232957>

※3 <http://ja.stackoverflow.com/questions/2979/>

※4 <http://hail2u.net/blog/webdesign/fixed-pseudo-element-on-mobile-safari-8.html>



探せばもっとたくさんのバグが  
見つかることでしょう

<https://www.google.co.jp/search?q=iOS+fixed+おかしい>

デスクトップブラウザも...

**transformプロパティで変形された要素には  
position: fixed;は効かない**

# DEMO

効く

[http://jsfiddle.net/o\\_ti/h5syu7dv/1/](http://jsfiddle.net/o_ti/h5syu7dv/1/)

効かない

[http://jsfiddle.net/o\\_ti/h5syu7dv/2/](http://jsfiddle.net/o_ti/h5syu7dv/2/)

バグなんだろう？

## CSS Transforms Module Level 1 - 6 The Transform Rendering Model

<http://www.w3.org/TR/css-transforms-1/#issue-ca2c412c>

### “ ISSUE 1

Is this effect on position: fixed necessary? If so, need to go into more detail here about why fixed positioned objects should do this, i.e., that it's much harder to implement otherwise. See Bug 16328.

## 【邦訳】 CSS Transforms Module Level 1 - 6. 変形の描画モデル

[http://www.hcn.zaq.ne.jp/\\_\\_\\_/WEB/css-transforms-ja.html#transform-rendering](http://www.hcn.zaq.ne.jp/___/WEB/css-transforms-ja.html#transform-rendering)

### “ 課題

これは固定位置（`position: fixed`）にも効果を及ぼす必要があるだろうか？ そうであれば、固定位置にされたオブジェクトがこれをすべき理由について詳細を述べる必要がある – その実装はずっと困難になるので。 Bug 16328

仕樣的にはバグというより問題未解決状態。

transformされた要素内のfixed値を規定通りにレンダリングさせるのはたいへん困難らしく、ブラウザベンダーが実装を進めていない。

最低限表示を壊さないようにposition: relative;としてレンダリングするようになっているのではないか？（憶測）

[http://jsfiddle.net/o\\_ti/h5syu7dv/4/](http://jsfiddle.net/o_ti/h5syu7dv/4/)



# 解決策

fixedの中に要素を作って  
それにtransformするなら問題ない

[http://jsfiddle.net/o\\_ti/h5syu7dv/3/](http://jsfiddle.net/o_ti/h5syu7dv/3/)

fixedした要素を含めた外側にtransformプロパティ  
があるとfixedが無効になるので、その逆ならOK

仕様と対応状況を理解すれば  
解決策を導くのも難しいことではない

**fixed**あるある

fixedしている要素がviewportから隠れていたら  
隠れた部分はページをいくらスクロールしても  
隠れたまま

たとえばページを印刷しても  
**viewport**から隠れている部分は一生出てこない

これは仕様通りの挙動である

- ずいぶん大きなモーダル
  - モーダル下部に閉じるボタンがあると隠れた時に押せない！モーダルを閉じられない！
- 巨大な入れ子メニューを擁するヘッダー
  - アコーディオンで展開されたリンクが隠れると、スクロールできないので永久に押せない

でもページを縮小したり  
スマホでもピンチアウトすれば  
見られるよ！

いちいち縮小と拡大を繰り返さないとい  
けないウェブサイト...

明日またアクセスしますか？



**“user-scalable=no”の前でも  
同じこと言えますか？**

それ、ユーザーの前でも  
見られるって言えんの？

仕様とブラウザ実装を知ったなら  
すべきこと・すべきでないことは  
おのずと明らかになる

viewportから意図せずに  
隠れてしまうfixedな要素は  
生み出してはならない

スマートフォンでは  
バグをたくさん抱えている

PCなら大丈夫と思っても  
タブレット対応してとか  
あとから言われると...

死

**fixedのない生活を！**

どうしても  
position: fixed;したい  
俺たちは



viewportより大きな要素をfixedさせたいけど  
内容物も全部見せたい場合にどうする？

# viewport unitの活用

viewport unitとは表示領域の大きさを  
基準とした相対的な単位のこと

**vw ... viewport width**

**vh ... viewport height**

**1vw = viewportの幅の1/100**

**100vw = viewportの幅の100/100**

viewportより大きくなる時だけ  
要素内がスクロールできればいい

viewportに収まるなら全部表出させたい

# できてないDEMO

[http://jsfiddle.net/o\\_ti/vrxwkbwk/](http://jsfiddle.net/o_ti/vrxwkbwk/)

# できてるDEMO

```
.menu {  
  position: fixed;  
  ...  
  max-height: 100vh;  
  overflow: auto;  
}
```

[http://jsfiddle.net/o\\_ti/vrxwkbwk/1/](http://jsfiddle.net/o_ti/vrxwkbwk/1/)

「いや～、全画面のメニューの中にも  
ヘッダー？みたいなのがあって  
それは絶対に隠したくないんだよね～」

あるあるですね

# calc()の活用



# calc()とは...

CSSのlengthデータ型をブラウザに四則演算させる関数で  
そのままwidthやfont-sizeなどのプロパティの値として  
指定できる

異なる単位も扱うことができ  
算出結果が割り切れない値でもよしなにしてくれる優れもの

```
.menu {  
    position: fixed;  
    ...  
}  
  
.menu__header {  
    height: 50px;  
    ...  
}  
  
.menu__list {  
    ...  
    height: calc(100vh - 50px);  
    overflow: auto;  
}
```

[http://jsfiddle.net/o\\_ti/vrxwkbwk/2/](http://jsfiddle.net/o_ti/vrxwkbwk/2/)

# calc()のブラウザ対応状況

Internet Explorer **9+** background-positionに使うとブラウザがクラッシュ！

---

Firefox **4+** -moz-が必要。プリフィックス無しは**16**から

---

Safari **6+** -webkit-が必要。プリフィックス無しは**6.1**から

---

Chrome **19+** -webkit-が必要。プリフィックス無しは**26**から

---

Opera **15+** いわゆるBlink版Operaからプリフィックス無しで使用可能。**Presto時代は未対応**

---

iOS Mobile Safari **6.1+** -webkit-必要。プリフィックス無しは**7.1**から

---

Android標準ブラウザ **4.4+** ただし**4.4.4**まで乗除ができない。Android Chromeでは40からプリフィックス無しで使用可能

現在の最新版でいえば、デスクトップブラウザは全てベンダープリフィックスなしで使用可能

スマートフォンはiOS Safari 7.1  
Android Chrome 40以上で  
プリフィックスなしで使用可能

viewport unitもcalc()も使えない場合

**IE8、Android 2.xではcalc()は未対応...**

可能なら、要素内でスクロールする前提の  
デザインをしてもらいましょう

高さのブレイクポイントを決めて、fixedする要素の  
高さが最低ラインまでにしないようにする

デザインで吸収できない場合は  
JavaScriptに助けてもらいましょう



モーダルな要素はfixedではなく absoluteにする  
&  
モーダル表出時にスクロール量に合わせて  
top値をJavaScriptで差し込む

absoluteな要素ならスクロールできるように  
なるので高さが画面より大きくなっても問題ない

- デザインを妥協するのか
  - 機能を妥協するのか
- 
- fixedする要素内にどんなものが入るのか
  - 本当にfixedにする必要があるのか

**チームでよく話し合って決めましょう**

オススメはfixedのない生活です

# まとめ

- fixedの仕様
- ブラウザの実装
- ブラウザのバグ
- fixedあるある
- 対策

おわりに

ウェブサイトに応用っぽいUIを提供できる  
`position: fixed;`は恰好良いし便利っぽい



しかしブラウザの対応状況は  
スマートフォンだとそこまで安定していない

**当該要素が画面から隠れる場合の対応は  
デザインや機能面で制約を増やすことも多い**

最新ブラウザで実装が安定して  
仕様が勧告になっても  
CSS Module Level 3プロパティーとの併用で  
挙動が変わることもある

CSS Module Level 4の登場で  
今後どうなるかわからないことばかり

W3Cの仕様もブラウザの実装も  
“生き物”である

**“CSSの基本は仕様の理解から”**

**今日これまでの発表でも  
知らなかった仕様はたくさんあったと思う**

今ウェブサイトを作ろうとしてCSSのことを  
調べると、仕様をたどって本当に詳しく書いて  
ある技術系のブログはとても少ない

こまかい仕様もちゃんと定義されているので  
ブログを見て「できた～」満足せずに  
自分で調べることが大事です



`position: fixed;`に限らず  
「CSSデキル」と胸を張って  
言えることを目指しましょう！

# ありがとうございました



Back to Basics

position: fixed; チョットデキル

## 謝辞

もんど氏

<http://momdo.s35.xrea.com/web-html-test/CSS3-ja/>

広瀬行夫氏

[http://www.hcn.zaq.ne.jp/\\_\\_\\_/](http://www.hcn.zaq.ne.jp/___/)