# 座標

要素を移動させるには、座標についてよく知っている必要があります。

ほとんどの JavaScript メソッドは２つの座標系のいずれかを扱います:

1. ウィンドウ(もしくは別のビューポート)の 上/左 を基準にします
2. ドキュメントの 上/左 を基準とします

違いを理解し、どのタイプがどこにあるかを理解することは重要です。

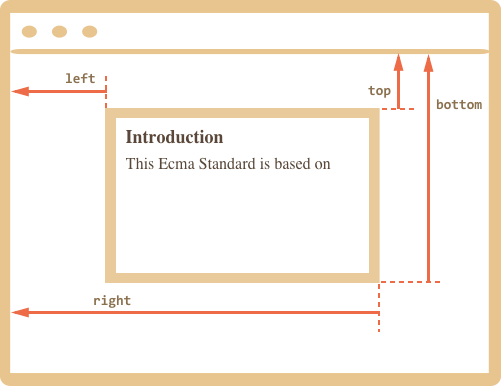
## [ウィンドウ座標: getBoundingClientRect](https://ja.javascript.info/coordinates" \l "ref-277)

ウィンドウ座標はウィンドウの左上端から始まります。

メソッド elem.getBoundingClientRect() はプロパティを持つオブジェクトとして elem に対するウィンドウ座標を返します。:

* top – 要素の上端の Y-座標,
* left – 要素の左端の X-座標,
* right – 要素の右端の X-座標,
* bottom – 要素の下端の Y-座標.

このようになります:



ウィンドウ座標はドキュメントのスクロールアウト部分を考慮せず、ウィンドウの左上端から計算されたものになります。

言い換えると、ページをスクロールするとき、要素は上下に移動し、そのウィンドウ座標は変わります。これはとても重要です。

ボタンをクリックしてウィンドウ座標を見てみてください:

もしページをスクロールすると、ボタン位置は代わり、ウィンドウ座標も同様に変わります。

また:

* 座標は小数の場合があります。それは正常で、内部的にはブラウザは計算のためにそれを使用します。私たちは style.position.left/top へ設定するときにそれらを丸める必要はありません。ブラウザで小数は問題ありません。
* 座標は負の値になる場合があります。例えば、ページが下にスクロールされ、elem の上端がウィンドウの上にある場合、elem.getBoundingClientRect().top は負の値になります。
* Chromeような一部のブラウザでは、結果 getBoundingClientRect にプロパティ width と height も追加します。減算することでもそれらを取得することは可能です: height=bottom-top, width=right-left。

**座標 右/下 は CSS プロパティとは異なります**

ウィンドウ座標と CSSの配置を比較すると、position:fixed との明らかな類似点があります – ビューポートを基準とした位置も同じです。

しかし、CSS では right プロパティは右端からの距離を意味し、bottom は – 下端からの距離です。

上の図を見た時、JavaScriptではそうでないことが分かります。すべてのウィンドウ座標は左上隅から数えられます。

## [elementFromPoint(x, y)](https://ja.javascript.info/coordinates" \l "ref-278)

document.elementFromPoint(x, y) の呼び出しは、ウィンドウ座標 (x, y) で最もネストされた要素を返します。

構文は次の通りです:

let elem = document.elementFromPoint(x, y);

例えば、下のコードは今ウィンドウの中央にある要素のタグを強調表示し、出力します。:

let centerX = document.documentElement.clientWidth / 2;

let centerY = document.documentElement.clientHeight / 2;

let elem = document.elementFromPoint(centerX, centerY);

elem.style.background = "red";

alert(elem.tagName);

ウィンドウ座標を使うので、要素は現在のスクロール位置に応じて異なります。

**ウィンドウ外の座標の場合、elementFromPoint は null を返します。**

メソッド document.elementFromPoint(x,y) は (x,y) が可視領域にある場合にのみ動作します。

もしある座標が負の値またはウィンドウの幅/高さを超えている場合、null を返します。

ほとんどの場合、このような振る舞いは問題ではありませんが、それを心に留めておく必要があります。

これは、それをチェックしない場合に発生する可能性のある典型的なエラーです:

let elem = document.elementFromPoint(x, y);

// 座標がウィンドウ外の場合、elem = null

elem.style.background = ''; // エラー!

## [position:fixed を用いる](https://ja.javascript.info/coordinates" \l "ref-279)

多くの場合、何かを配置するために座標を必要とします。CSS ではビューポートを基準として要素を配置するために、left/top (または right/bottom) と一緒に position:fixed を使います。

私たちは、getBoundingClientRect を使って要素の座標を取得し、その近くに何かを表示することができます。

例えば、下の関数 createMessageUnder(elem, html) は elem の下にメッセージを表示します。:

let elem = document.getElementById("coords-show-mark");

function createMessageUnder(elem, html) {

// メッセージ要素の作成

let message = document.createElement('div');

// ここでは、スタイルにCSSクラスを使う方が良いです

message.style.cssText = "position:fixed; color: red";

// 座標の設定, "px" を忘れないこと!

let coords = elem.getBoundingClientRect();

message.style.left = coords.left + "px";

message.style.top = coords.bottom + "px";

message.innerHTML = html;

return message;

}

// 使用例:

// ドキュメント上に５秒間追加します

let message = createMessageUnder(elem, 'Hello, world!');

document.body.append(message);

setTimeout(() => message.remove(), 5000);

実行するにはボタンをクリックしてください:

id=“coords-show-mark” のボタン, この下にメッセージが現れます

このコードを変更して、メッセージを左、右、下に表示したり、“フェードイン” するための CSS アニメーションを適用することができます。私たちは要素のすべての座標とサイズを知っているので簡単にできます。

しかし、重要な点に注意してください: ページがスクロールされたとき、メッセージはボタンから離れていきます。

理由は明らかです: メッセージ要素は position:fixed に依存しているので、ページがスクロールしている間、ウィンドウの同じ場所にい続けます。

変更するためには、ドキュメントベースの座標を使い、position:absolute を使う必要があります。

## [ドキュメント座標](https://ja.javascript.info/coordinates" \l "ref-280)

ドキュメント相対座標は、ウィンドウではなくドキュメントの左上端から始めます。

CSS では、ウィンドウ座標は position:fixed に対応する一方、ドキュメント座標は position:absolute に似ています。

position:absolute と top/left を使うことで、ドキュメント上の特定の場所に何かを置くことができます。なので、ページのスクロール時にそこに残ることができます。しかし、最初に正しい座標が必要です。

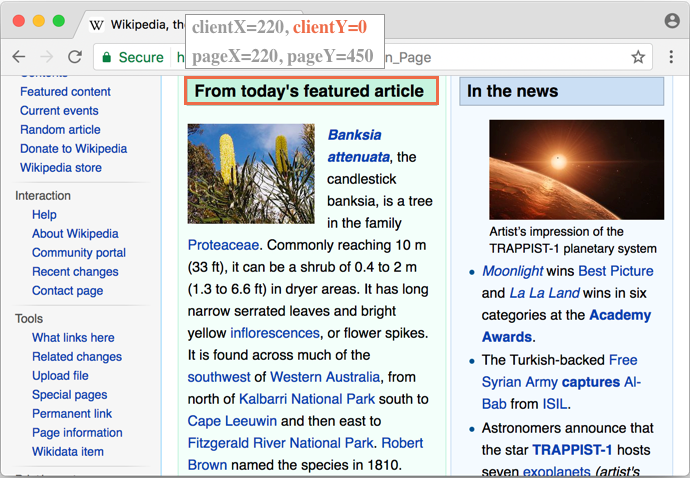
分かりやすくするために、ウィンドウ座標 (clientX,clientY) とドキュメント座標 (pageX,pageY) を呼び出します。

ページがスクロールされていないとき、ウィンドウ座標とドキュメント座標はまったく同じです。それらのゼロの点も一致します:



そして、スクロールをすると、(clientX,clientY) が変わります。なぜなら、それらはウィンドウに相対的なためです。しかし、(pageX,pageY) は同じままです。

これは縦スクロール後の同じページです:



* 要素は今ウィンドウの上部にあるので、ヘッダー "From today's featured article" の clientY は 0 になりました。
* 水平スクロールをしなかったため、clientX は変わりませんでした。
* 要素の pageX と pageY 座標は依然として同じです。なぜなら、それらはドキュメントに相対的だからです。

## [ドキュメント座標の取得](https://ja.javascript.info/coordinates" \l "ref-281)

要素のドキュメント座標を取得するための標準メソッドはありません。しかし、簡単に書けます。

２つの座標系は次の式で繋がれます:

* pageY = clientY + ドキュメントのスクロールアウトした垂直部分の高さ
* pageX = clientX + ドキュメントのスクロールアウトした水平部分の幅

関数 getCoords(elem) は elem.getBoundingClientRect() からウィンドウ座標を取り、それらに現在のスクロールを加えます。:

// 要素のドキュメント座標を取得

function getCoords(elem) {

let box = elem.getBoundingClientRect();

return {

top: box.top + pageYOffset,

left: box.left + pageXOffset

};

}

## [サマリ](https://ja.javascript.info/coordinates" \l "ref-282)

ページ上に任意の点は座標を持っています:

1. ウィンドウに相対的 – elem.getBoundingClientRect()
2. ドキュメントに相対的 – elem.getBoundingClientRect() + 現在のページスクロール

ウィンドウ座標は position:fixed と一緒に使用するのが賢明で、ドキュメント座標は position:absolute と上手くやります。

どちらの座標系も “長所” と “短所” を持っており、CSS の position absolute と fixed のように、どちらか一方が必要なときがあります。

## [タスク](https://ja.javascript.info/coordinates#tasks)

### [フィールドのウィンドウ座標を見つける](https://ja.javascript.info/coordinates" \l "ref-286)

重要性: 5

下の iframe では、緑の “フィールド” を持つドキュメントがあります。

矢印でポイントされた角のウィンドウ座標を見つけるために、JavaScript を使用してください。

便宜のために、ドキュメント上に実装している小さな機能があります。任意の場所をクリックすると、そこの座標を表示します。

あなたのコードは次のウィンドウ座標を取得するために DOM を使う必要があります:

1. 左上の外部の角(それはシンプルです)
2. 右下の外部の角(これもシンプルです)
3. 左上の内部の角(少し難しいです)
4. 右下の内部の角(いくつか方法があり、１つを選んでください)

あなたが計算する座標は、マウスクリックで返却されるものと同じであるべきです。

P.S. コードは要素が別のサイズやボーダーの場合でも動作する必要があります。任意の固定値で束縛しないでください。

[タスクのためのサンドボックスを開く](https://plnkr.co/edit/qcWJdTiRvCzUyN7S?p=preview)

解答

### [要素の近くにノート(メモ)を表示する](https://ja.javascript.info/coordinates" \l "ref-287)

重要性: 5

要素 anchor の 上 ("top"), 右 ("right") または下 ("bottom") のいずれかにある position に応じて elem を配置する関数 positionAt(anchor, position, elem) を作成してください。

それを使って、クラス "note" を持つ要素と、anchor の近くの指定された位置にテキスト html を表示する関数 showNote(anchor, position, html) を作成します。

次のようなノートを表示します:

P.S. ノートはこのタスクでは position:fixed を持たなければなりません。

[タスクのためのサンドボックスを開く](https://plnkr.co/edit/ZuHCpkNaBjCuWiGD?p=preview)

解答

### [要素の近くにノートを表示する(absolute)](https://ja.javascript.info/coordinates" \l "ref-288)

重要性: 5

ノートが position:fixed の代わりに position:absolute を使うよう、 [前のタスク](https://ja.javascript.info/task/position-at) の解決策を修正してください。

それはページスクロール時に、要素から “離れていく” のを防ぎます。

そのタスクの解決策を出発点にしてください。スクロールをテストするために、スタイル <body style="height: 2000px"> を追加してください。

解答

### [ノートを内部に配置する (absolute)](https://ja.javascript.info/coordinates" \l "ref-289)

重要性: 5

前のタスク [要素の近くにノートを表示する(absolute)](https://ja.javascript.info/task/position-at-absolute) を拡張してください: 関数 positionAt(anchor, position, elem) に、 anchor の内側に elem を挿入するよう教えてください。

position のための新しい値:

* top-out, right-out, bottom-out – 前と同じように動作し、elem を anchor の上/右/下に挿入します。
* top-in, right-in, bottom-in – anchor の内側に elem を挿入します。: 上/右/下端に取り付けます。

例:

// blockquote の上にノートを表示します

positionAt(blockquote, "top-out", note);

// blockquote の内側の先頭にノートを表示します

positionAt(blockquote, "top-in", note);

結果:

ソースコードとして、タスク [要素の近くにノートを表示する(absolute)](https://ja.javascript.info/task/position-at-absolute) の解決策を使ってください。

解答