



CSS 3



アジェンダ

- 1.CSS2.1→CSS3 追加されたセレクタ
 - 2.CSS2.1→CSS3 追加されたプロパティ
 - 3.プログレッシブ・エンハンスメント
 - 4.エフェクト
 - 5.アニメーション
-

CSS2.1→CSS3 追加されたセレクタ

- シンプルセレクタ(属性セレクタ)
 - [attr^="val"]
 - [attr\$="val"]
 - [attr*="val"]
- 結合子
 - A ~ B

CSS2.1→CSS3 追加されたセレクタ

- シンプルセレクタ(疑似クラス)
 - :target
 - :enabled
 - :disabled
 - :checked
 - :indeterminate
 - :root
 - :first-child
 - :last-child
 - :nth-child(...)
 - :nth-last-child(...)
 - :only-child
 - :first-of-type
 - :last-of-type
 - :nth-of-type(...)
 - :nth-last-of-type(...)
 - :only-of-type
 - :empty
 - :not(~)
-

CSS2.1→CSS3 追加されたセレクタ

意外に多い(特に疑似クラスが増えた)

iOS (4.3)

Firefox (4.0)

Chrome (11)

Internet Explorer (9)

Android (2.3.3)

Safari (5.0.5)

Opera (11.10)

CSS2.1→CSS3 追加されたセレクタ

インターフェースセレクタ

セレクタ	適用先
:valid	入力値が正しい場合に適用
:invalid	入力値が正しくない場合に適用
:required	入力が必要の場合に適用
:optional	入力が必要ではない場合に適用
:in-range	入力値が範囲内の場合に適用
:out-range	入力値が範囲外の場合に適用
:read-only	変更できない要素に適用
:read-write	変更できる要素に適用
:default	標準で選択された要素に適用

CSS2.1→CSS3 追加されたセレクタ

演習

```
:enabled {  
  background-color: #FFEFD5;  
}  
:disabled {  
  background-color: #D3D3D3;  
}
```

```
<p><input type="text" value="hoge" /></p>  
<p><input type="text" value="huga"  
  disabled="disabled" /></p>
```

CSS2.1→CSS3 追加されたセレクタ

演習

```
table, tr, th, td {  
  border: solid 1px gray;  
}  
table tr th {  
  background-color: #0FF;  
}  
table tr:nth-child(even) td {  
  background-color: #F5F5F5;  
}  
table tr:nth-child(odd) td {  
  background-color: #E0FFFF;  
}
```

```
<table>  
<tr><th>都道府県</th><th>県庁所  
在地</th></tr>  
<tr><td>茨城県</td><td>水戸市  
</td></tr>  
<tr><td>栃木県</td><td>宇都宮市  
</td></tr>  
<tr><td>群馬県</td><td>前橋市  
</td></tr>  
<tr><td>埼玉県</td><td>さいたま  
市</td></tr>  
<tr><td>千葉県</td><td>千葉市  
</td></tr>  
<tr><td>東京都</td><td>新宿区  
</td></tr>  
<tr><td>神奈川県</td><td>横浜市  
</td></tr>  
</table>
```


メディアクエリ

- メディアタイプそのものはCSS2からあった
 - @media "メディアタイプ" { ... }
 - <http://www.htmq.com/csskihon/009.shtml>
- CSS3ではさらに特性が指定できるようになった
 - @media "メディアタイプ" and ("特性") { ... }
 - <http://robertnyman.com/2010/09/09/css3-media-queries-and-creating-adaptive-layouts/>

メディアクエリ

よく使いそうな特性

- width
 - height
 - device-width
 - device-height
 - orientation
 - portrait
 - landscape
 - resolution
 - 単位はdpi (または dpcm)
 - min- や max- のプレフィックスで最小/最大の指定ができる
-

メディアクエリ

演習

先ほどのtableを使用したHTMLにCSSを追加する

```
body {  
  background-color: pink;  
}  
@media screen and (min-width: 800px) {  
  body {  
    background-color: green;  
  }  
}
```

CSS2.1→CSS3 追加されたプロパティ

- 角丸
- Webフォント
- コンテンツの挿入・置換

※もちろん一部です

CSS3 追加されたプロパティ 角丸

角丸

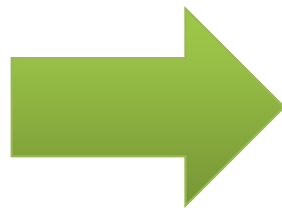
プロパティ	意味
border-radius: 角丸の半径	角を丸くする
border-radius: 横方向の半径 / 縦方向の半径	角を楕円状に丸くする
border-top-left-radius	左上の角を丸くする
border-top-right-radius	右上の角を丸くする
border-bottom-left-radius	左下の角を丸くする
border-bottom-right-radius	右下の角を丸くする

CSS3 追加されたプロパティ 角丸

演習

半径は10pxで作成しましょう

都道府県	県庁所在地
茨城県	水戸市
栃木県	宇都宮市
群馬県	前橋市
埼玉県	さいたま市
千葉県	千葉市
東京都	新宿区
神奈川県	横浜市



都道府県	県庁所在地
茨城県	水戸市
栃木県	宇都宮市
群馬県	前橋市
埼玉県	さいたま市
千葉県	千葉市
東京都	新宿区
神奈川県	横浜市

CSS3 追加されたプロパティ Webフォント

Webフォント

- @font-face { ... } でオンラインからフォントを読み込んで表示させることができます
- WOFF (.woff)形式がWebフォント用のファイル
 - 全てのブラウザが対応できていないのでTrueType (.ttf)またはOpenType (.ptf)も一緒に用意する

CSS3 追加されたプロパティ Webフォント

Webフォントを公開しているサイト

Font Squirrel

<http://www.fontsquirrel.com/>

Google Web Fonts

<http://www.google.com/webfonts#ChoosePlace:select>

CSS3 追加されたプロパティ Webフォント

演習

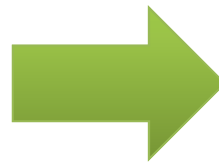
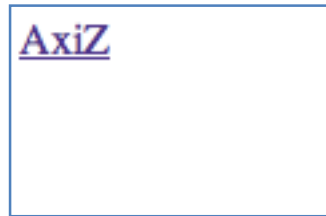
```
<link href="http://fonts.googleapis.com/css?family=Leckerli+One"
rel="stylesheet" type="text/css">
<style type="text/css">
span {
  font-family: "Leckerli One", cursive;
}
</style>
<body>
Hello, World <span>Hello, World</span>
</body>
```

Google Web Fontsの場合は
@font-faceはいらない。

コンテンツの挿入・置換

演習

文字列リンクを画像に置換する



`AxiZ`

画像のURL

`=> http://www.axiz.co.jp/img/axiz_run_your_future.gif`

コンテンツの挿入・置換

解答(CSS2.1まで)

```
#link_axiz {  
    background-image:  
        url(http://www.axiz.co.jp/img/axiz_run_your_future.gif);  
    background-repeat: no-repeat;  
    display: inline-block;  
    width: 105px;  
    height: 70px;  
    text-indent: -9999px;  
}
```

コンテンツの挿入・置換

CSS3では

```
#link_axiz {  
  content:  
    url(http://www.axiz.co.jp/img/axiz_run_your_future.gif);  
}
```

コンテンツの挿入・置換

contentプロパティの変更

- contentプロパティはCSS2.1までは :before, :after の疑似要素でしか使えませんでした(CSS3では ::before, ::after と記述します)
- CSS3からは全要素でcontentプロパティが使用できるようになりました

```
.required:after {  
  content: "*";  
  color: red;  
}
```

```
<span class="required">お名前</span>  
<input type="text" />
```

プログレッシブ・エンハンスメント

- クロスブラウザはもう古い
 - 同じ"見え方"を目指すのは開発者のエゴ
 - 情報さえちゃんとあるなら、
“見え方”は異なってもかまわない
- TwitterやYoutubeではすでに取り入れられている

プログレッシブ・エンハンスメント

Youtubeの例

グラデーション非対応

世界最大の動画共有コミュニティに参加しよう!

アカウントを作成 ›

アカウントをお持ちですか? ログイン

グラデーション対応

世界最大の動画共有コミュニティに参加しよう!

アカウントを作成 ›

アカウントをお持ちですか? ログイン

プログレッシブ・エンハンスメント

Twitterの例



Twitter login form with sharp corners. The form includes fields for 'ユーザー名' (Username), 'パスワード' (Password), and a 'ログイン' (Login) button. Below these is a checkbox for '保存する' (Remember me) and a link 'パスワードを忘れた場合はこちら' (Click here if you forgot your password). A message 'Twitter は初めて? 早速、登録しよう!' (Twitter is new? Sign up now!) is followed by fields for '名前' (Name), 'メールアドレス' (Email address), and 'パスワード' (Password). A yellow '新規登録' (Sign up) button is at the bottom right.

角丸非対応

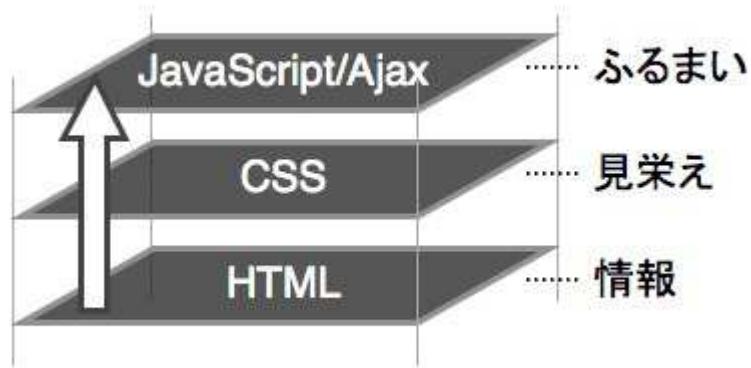


Twitter login form with rounded corners. The form includes fields for 'ユーザー名' (Username), 'パスワード' (Password), and a 'ログイン' (Login) button. Below these is a checkbox for '保存する' (Remember me) and a link 'パスワードを忘れた場合はこちら' (Click here if you forgot your password). A message 'Twitter は初めて? 早速、登録しよう!' (Twitter is new? Sign up now!) is followed by fields for '名前' (Name), 'メールアドレス' (Email address), and 'パスワード' (Password). A yellow '新規登録' (Sign up) button is at the bottom right.

角丸対応

サイト構築の考え方

- 1.一番大事なのはHTML。情報の論理構造化
- 2.次にCSS。高機能なクライアントにはよりリッチなデザインを提供する
- 3.最後にJavaScript。高機能なクライアントにはより豊かなユーザーエクスペリエンスを提供する



エフェクト

- アルファチャンネル
 - ドロップシャドウ
 - グラデーション
 - 2Dトランスフォーム
 - 3Dトランスフォーム
-

エフェクト アルファチャンネル

アルファチャンネル

```
p {  
  color: rgba(0, 0, 0, 0.5);  
  background-color: rgba(0, 0, 255, 0.4);  
}  
img {  
  opacity: 0.6;  
}
```

Hello, AlphaChannel



```
<p>Hello, AlphaChannel</p>  

```

エフェクト ドロップシャドウ

ドロップシャドウ

- box-shadowとtext-shadowの2つがある
 - Webkitはbox-shadowにベンダープレフィックスをつける
- 影はぼかしたり、応用でグロー効果をつけたりできる

エフェクト ドロップシャドウ

演習

```
span {  
  background-color: blue;  
  border-radius: 5px;  
  box-shadow: 5px 5px 5px 0 lightblue;  
  -webkit-box-shadow: 5px 5px 5px 0 lightblue;  
  
  font-size: 30px;  
  padding: 5px;  
  text-shadow: 5px 5px 5px gray;  
}
```

横オフセット 縦オフセット ブラー スプレッド 色

```
<span>Some Title</span>
```

エフェクト グロー効果

グロー効果として

```
span {  
  background-color: blue;  
  border-radius: 5px;  
  box-shadow: 5px 5px 5px 0 lightblue;  
  -webkit-box-shadow: 5px 5px 5px 0 lightblue;
```

```
  font-size: 30px;  
  padding: 5px;  
  text-shadow: 0 0 5px white,  
               0 0 5px white,  
               0 0 5px white,  
               0 0 5px white;
```

```
}
```

```
<span>Some Title</span>
```

4重にしてグロー効果を強化

エフェクト グラデーション

グラデーション

- 線形グラデーションと円形グラデーションがある
- Webkitだけ書き方が異なる
 - つまり2パターン併記する必要がある
 - ただChromeやMac版Safari5.1では標準の表記も一部対応している
- W3C標準表記の方が書き方が簡潔。
またWebkit固有とは100%互換ではない

エフェクト 線形グラデーション1

演習

角度(deg)

```
span {  
  background: -moz-linear-gradient(0, #F80 0%, #FF0 100%);  
  background: -webkit-linear-gradient(0, #F80 0%, #FF0 100%);  
  background: -o-linear-gradient(0, #F80 0%, #FF0 100%);  
  background: -ms-linear-gradient(0, #F80 0%, #FF0 100%);  
  background: -webkit-gradient(linear, left top, right top,  
    from(#F80), to(#FF0));  
  border-radius: 5px;  
  font-size: 30px;  
  padding: 5px;  
}  
<span>Some Title</span>
```


エフェクト 線形グラデーション2

演習

```
span {  
  background: -moz-linear-gradient(  
    0, #F00 0%, #F80 33%, #FF0 66%, #0F0 100%);  
  background: -webkit-linear-gradient(  
    0, #F00 0%, #F80 33%, #FF0 66%, #0F0 100%);  
  background: -o-linear-gradient(  
    0, #F00 0%, #F80 33%, #FF0 66%, #0F0 100%);  
  background: -ms-linear-gradient(  
    0, #F00 0%, #F80 33%, #FF0 66%, #0F0 100%);  
  background: -webkit-gradient(linear, left top, right top,  
    from(#F00), color-stop(#F80, 33%), color-stop(#FF0, 66%), to(#0F0));  
  border-radius: 5px;  
  font-size: 30px;  
  padding: 5px;  
}  
<span>Some Title</span>
```

エフェクト 円形グラデーション

演習

```
span {  
  background: -moz-radial-gradient(  
    15% 50%, circle closest-corner, #F80 0%, #FF0 100%);  
  background: -webkit-radial-gradient(  
    15% 50%, circle closest-corner, #F80 0%, #FF0 100%);  
  background: -o-radial-gradient(  
    15% 50%, circle closest-corner, #F80 0%, #FF0 100%);  
  background: -ms-radial-gradient(  
    15% 50%, circle closest-corner, #F80 0%, #FF0 100%);  
  background: -webkit-gradient(  
    radial, 15% 50%, 0, 15% 50%, 30, from(#F80), to(#FF0));  
  border-radius: 5px;  
  font-size: 30px;  
  padding: 5px;  
}  
<span>Some Title</span>
```

2Dトランスフォーム

- 回転
 - rotate()
- 拡大・縮小
 - scale() / scaleX() / scaleY()
- 移動
 - translate() / translateX() / translateY()
- スキュー(シアー)
 - skew() / skewX() / skewY()
- 変換マトリクス(変換行列)
 - matrix()

エフェクト 回転

演習

```
span {  
  -moz-transform: rotate(10deg);  
  -webkit-transform: rotate(10deg);  
  -o-transform: rotate(10deg);  
  -ms-transform: rotate(10deg);  
  background: orange;  
  border-radius: 5px;  
  display: inline-block;  
  font-size: 30px;  
  padding: 5px;  
}  
<span>Some Title</span>
```

Webページに配置した要素は左上を原点(0, 0)としますが、各変形処理を設定すると要素の(50%, 50%)の点が原点として処理が適用されます。

本当は必要ないはずだが、書かないと適用されない。

エフェクト 拡大縮小

演習

```
span {  
  -moz-transform: rotate(10deg) scale(0.5, 2);  
  -webkit-transform: rotate(10deg) scale(0.5, 2);  
  -o-transform: rotate(10deg) scale(0.5, 2);  
  -ms-transform: rotate(10deg) scale(0.5, 2);  
  background: orange;  
  border-radius: 5px;  
  display: inline-block;  
  Font-size: 30px;  
  padding: 5px;  
}
```

rotateは要素のもつローカル座標系を傾けます。
scaleは要素のもつローカル座標系の1目盛りの大きさが変化します。

エフェクト 移動

演習

```
span {  
  -moz-transform: rotate(10deg) scale(0.5, 2) translate(100px,  
    100px);  
  -webkit-transform: rotate(10deg) scale(0.5, 2)  
    translate(100px,100px);  
  -o-transform: rotate(10deg) scale(0.5, 2) translate(100px, 100px);  
  -ms-transform: rotate(10deg) scale(0.5, 2)  
    translate(100px, 100px);  
  background: orange;  
  border-radius: 5px;  
  display: inline-block;  
  font-size: 30px;  
  padding: 5px;  
}
```

実は・・・scale()とtranslate()
の順番を変える
とスタイルも変わります。

エフェクト スキュー (シアー)

演習

```
span {  
  -moz-transform: skewY(30deg);  
  -webkit-transform: skewY(30deg);  
  -o-transform: skewY(30deg);  
  -ms-transform: skewY(30deg);  
  background: orange;  
  border-radius: 5px;  
  display: inline-block;  
  font-size: 30px;  
  padding: 5px;  
}
```


rotateが座標系を傾ける処理
だったのに対して
skewは座標系を**歪ませる**処理
をする。

エフェクト 変換マトリクス (変換行列)


かなり数学よりな話しになるので飛ばします。
線型代数とか好きな人は好きになれるかも。

<http://www.useragentman.com/blog/2011/01/07/css3-matrix-transform-for-themathematicallychallenged/>

2Dトランスフォームの原点



```
span {
  display: inline-block;
  border-radius: 5px;
  background-color: green;
  font-size: 30px;
  padding: 5px;
}
#rotate-target {
  -moz-transform: rotate(30deg);
  -webkit-transform: rotate(30deg);
  -o-transform: rotate(30deg);
  -ms-transform: rotate(30deg);
  background: orange;
}
```



```
span {
  display: inline-block;
  border-radius: 5px;
  background-color: green;
  font-size: 30px;
  padding: 5px;
}
#rotate-target {
  -moz-transform: rotate(30deg);
  -webkit-transform: rotate(30deg);
  -o-transform: rotate(30deg);
  -ms-transform: rotate(30deg);
  -moz-transform-origin: 0% 100%;
  -webkit-transform-origin: 0% 100%;
  -o-transform-origin: 0% 100%;
  -ms-transform-origin: 0% 100%;
  background: orange;
}
```

Some Title

Some Title

3Dトランスフォーム

- 回転
 - rotate() / rotate3d() / rotateX() / rotateY() / rotateZ()
- 拡大・縮小
 - scale() / scale3d() / scaleX() / scaleY() / scaleZ()
- 移動
 - translate() / translate3d() / translateX() / translateY() / translateZ()
- スキュー(シアー)
 - skew() / skewX() / skewY()
- 変換マトリクス
 - matrix() / matrix3d()
- 透視投影
 - perspective()

3Dトランスフォーム

3Dになっても考え方は一緒です。
ただZ軸が増えるので、ちょっと複雑になります。

```
span {  
  -moz-transform: rotateY(60deg);  
  -webkit-transform: rotateY(60deg);  
  -o-transform: rotateY(60deg);  
  -ms-transform: rotateY(60deg);  
  background: orange;  
  border-radius: 5px;  
  display: inline-block;  
  font-size: 30px;  
  padding: 5px;  
}
```

3Dトランスフォーム



何かイメージと違う・・・

3Dトランスフォーム

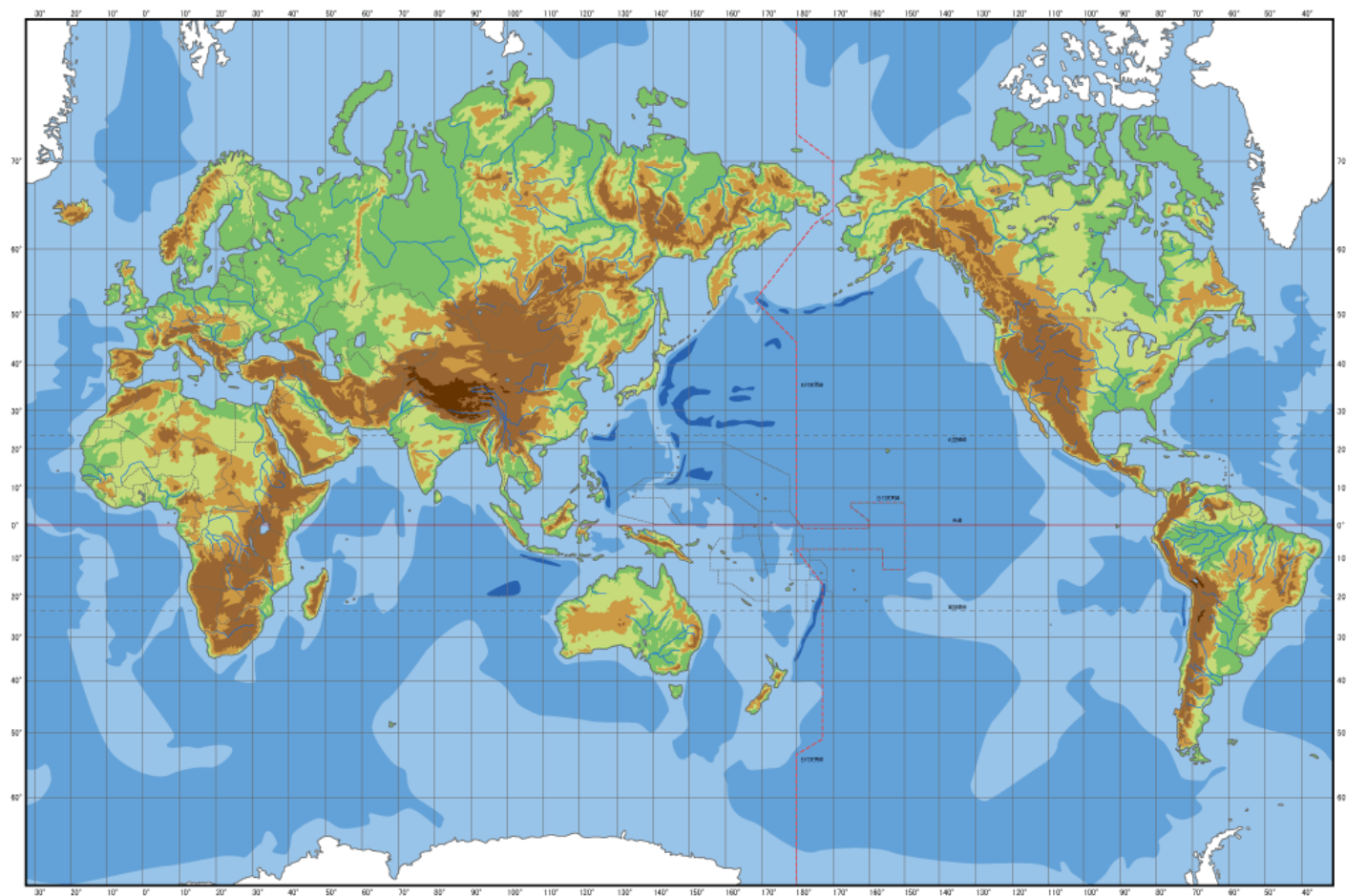
実はこう書く。

```
span {  
  -moz-transform: perspective(200) rotateY(60deg);  
  -webkit-transform: perspective(200) rotateY(60deg);  
  -o-transform: perspective(200) rotateY(60deg);  
  -ms-transform: perspective(200) rotateY(60deg);  
  background: orange;  
  border-radius: 5px;  
  display: inline-block;  
  font-size: 30px;  
  padding: 5px;  
}
```



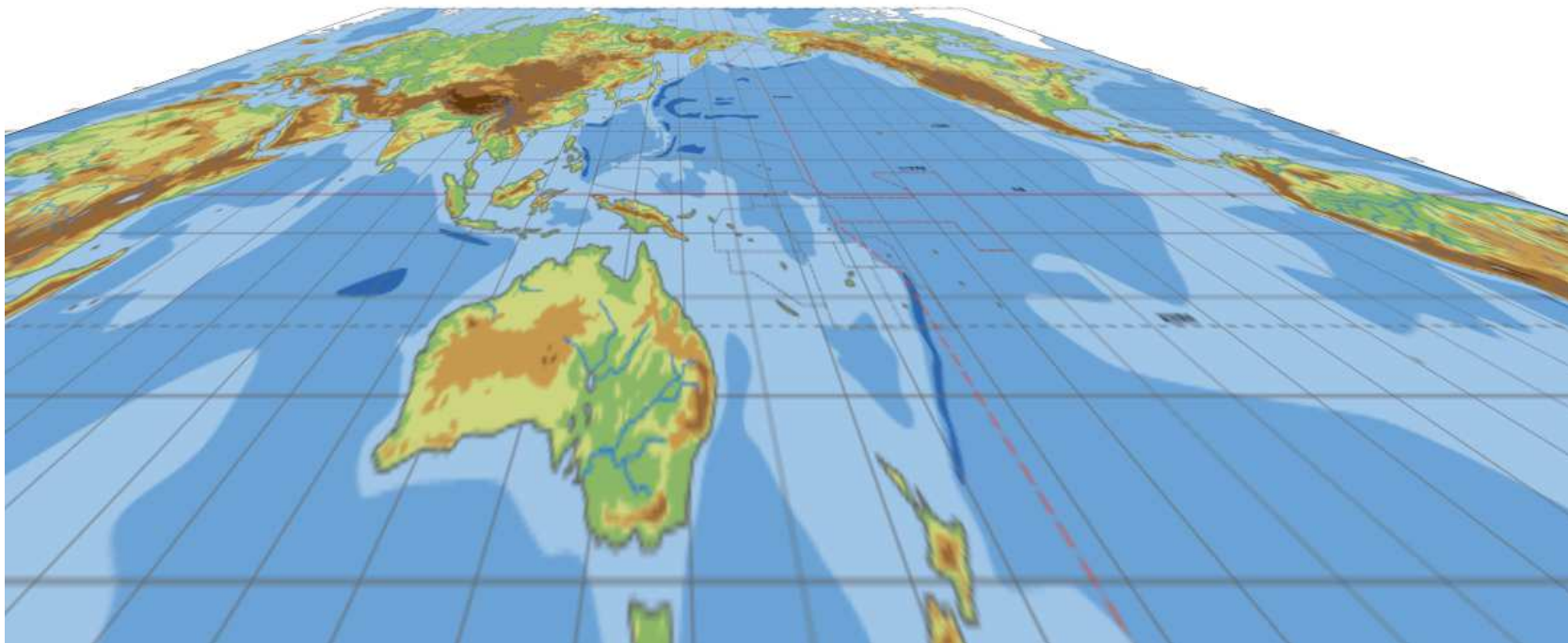
3Dトランスフォーム

3Dトランスフォームの例 before



3Dトランスフォーム

after



アニメーション

- トランジション
 - 始点と終点の2点間をアニメーション変化
- アニメーション
 - キーフレームを利用したアニメーション変化

アニメーション トランジション 1

5段階に分けて、
トランジションの適用方法をみていきます。

1

```
span {  
  font-size: 24px;  
  background-color: skyblue;  
}  
span:hover {  
  font-size: 36px;  
  background-color: yellow;  
}
```

Hello, Animation!

アニメーション トランジション 2

2

```
span {  
  font-size: 24px;  
  background-color: skyblue;  
  -webkit-transition: 1s;  
  -moz-transition: 1s;  
  -o-transition: 1s;  
}  
span:hover {  
  font-size: 36px;  
  background-color: yellow;  
}
```

Hello, Animation!

アニメーション トランジション 3

3

```
span {  
  font-size: 24px;  
  background-color: skyblue;  
  -webkit-transition: 1s background-color;  
  -moz-transition: 1s background-color;  
  -o-transition: 1s background-color;  
}  
span:hover {  
  font-size: 36px;  
  background-color: yellow;  
}
```

```
<span>Hello, Animation!</span>
```

アニメーション トランジション 4

4

```
span {  
  font-size: 24px;  
  background-color: skyblue;  
  -webkit-transition: 1s 3s;  
  -moz-transition: 1s 3s;  
  -o-transition: 1s 3s;  
}  
span:hover {  
  font-size: 36px;  
  background-color: yellow;  
}
```

```
<span>Hello, Animation!</span>
```

アニメーション トランジション 5

5

```
span {  
  font-size: 24px;  
  background-color: skyblue;  
  -webkit-transition: 1s font-size, 1s background-color 1s;  
  -moz-transition: 1s font-size, 1s background-color 1s;  
  -o-transition: 1s font-size, 1s background-color 1s;  
}  
span:hover {  
  font-size: 36px;  
  background-color: yellow;  
}
```

```
<span>Hello, Animation!</span>
```

アニメーション キーフレーム

アニメーションでは、
キーフレームを定義することで
よりきめ細やかなアニメーションを
表現する事ができます。

現状Webkitのみ実装している(iOS、Android含む)。

アニメーション キーフレーム 1

1

```
@-webkit-keyframes color-change {  
  0% {  
    background-color: skyblue;  
  }  
  50% {  
    background-color: pink;  
  }  
  100% {  
    background-color: yellow;  
  }  
}  
  
span {  
  font-size: 24px;  
  background-color: gray;  
  -webkit-animation: color-change 5s;  
}
```

Hello, Animation!

アニメーション キーフレーム 2

2

```
@-webkit-keyframes color-change {  
  0% {  
    background-color: skyblue;  
    opacity: 0;  
  }  
  50% {  
    background-color: pink;  
  }  
  100% {  
    background-color: yellow;  
    opacity: 1;  
  }  
}  
span {  
  font-size: 24px;  
  background-color: gray;  
  -webkit-animation: color-change 5s;  
}
```


アニメーション キーフレーム 3

3

```
@-webkit-keyframes color-change {  
  0% {  
    background-color: skyblue;  
    opacity: 0;  
  }  
  50% {  
    background-color: pink;  
  }  
  100% {  
    background-color: yellow;  
    opacity: 1;  
  }  
}  
span {  
  font-size: 24px;  
  background-color: gray;  
  -webkit-animation: color-change 5s 3s;  
}
```

アニメーション キーフレーム 4

4

```
@-webkit-keyframes color-change {  
  0% {  
    background-color: skyblue;  
    opacity: 0;  
  }  
  50% {  
    background-color: pink;  
  }  
  100% {  
    background-color: yellow;  
    opacity: 1;  
  }  
}  
span {  
  font-size: 24px;  
  background-color: gray;  
  -webkit-animation: color-change 5s infinite;  
}
```

アニメーション キーフレーム 5

5

```
@-webkit-keyframes color-change {
  0% {
    background-color: skyblue;
  }
  50% {
    background-color: pink;
  }
  100% {
    background-color: yellow;
  }
}

@-webkit-keyframes text-opacity {
  0% {
    color: rgba(0, 0, 0, 0);
  }
  100% {
    color: rgba(0, 0, 0, 1);
  }
}

span {
  font-size: 24px;
  background-color: gray;
  -webkit-animation: color-change 5s infinite, text-opacity 10s 2;
}
```