Lesson4

02 CSS Gridを使ったレイアウト

90 min

THEME テーマ

CSS Gridと呼ばれる方法でレイアウトをしてみましょう。Flexboxやinline-blockと違い、行と列を使って二次元的に要素を配置をする方法です。IE11への対応はサポートする記述が必要です（IE10以前は非対応）。

CSS Gridとは

CSS Grid Layout Module（以下「CSS Grid」）とは、「グリッド」と呼ばれる見えない罫線を生成し、マス目を使ってレイアウトするシステムです。flexboxでも同じようなレイアウトを作れますが、CSS Gridは要素を二次元的に配置することができ、写真ギャラリーなどの配置も簡単に作ることができます。

図1 HTMLとGridを書く前段階のCSS

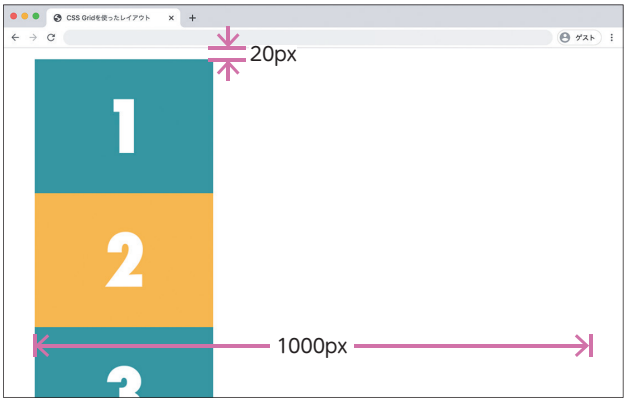
HTML

```
<ul class="container">
    <li class="item"><img src="img/1.png" alt="1"></li>
    <li class="item"><img src="img/2.png" alt="2"></li>
    <li class="item"><img src="img/3.png" alt="3"></li>
    <li class="item"><img src="img/4.png" alt="4"></li>
    <li class="item"><img src="img/5.png" alt="5"></li>
    <li class="item"><img src="img/6.png" alt="6"></li>
</ul>
```

CSS

```
/* 全体の設定・事前準備 */
*{
    box-sizing:border-box;
    margin: 0;
    padding: 0;
}
ul{list-style: none;}
img{
    display: block;
    max-width: 100%;
}
.container{
    width: 1000px;
    margin: 20px auto;
}
```

ブラウザ表示　Gridを使う前はただ縦に並んでいる



なお、CSS Gridは、現在はほとんどの最新ブラウザでサポートされていますが、IE11ではベンダープレフィックスや機能を補助する記述が必要です。

グリッドを作る

本節では番号の書かれた画像を行と列の2方向から指定して配置していきます。図1はベースとなるHTML/CSSです。グリッドはテーブルのように列と行で構成します。grid-template-columnsプロパティとgrid-template-rowsプロパティを使って、箱の数と大きさを指定します。grid-template-columnsは、数値を書いた数だけ横に列が増え、grid-template-rowsでは数値を書いた数だけ下に行が増えていきます。図2の記述では、320pxの列が3つと240pxの行が2つ生成されます。

なお、IE11には効かないため、-ms-grid-columnsと-ms-grid-rowsというプロパティも併記します。接頭辞「-ms-」だけでなくプロパティ名も異なっていることに注意しましょう。

アイテム同士の間隔を決めるには、marginなどは使わず、コンテナにgrid-gapプロパティで指定します。

POINT

古いブラウザに対応させるための接頭辞を「ベンダープレフィックス」といいます。ここではIE11に対応させるため「-ms-」というベンダープレフィックスを記述しています。

図2 グリッドを作る

CSS ※青字はIE11用の記述

```
/*Gridの設定*/
.container{
    display: -ms-grid;
    display: grid;
    -ms-grid-columns: 320px 20px 320px 20px 320px;
    grid-template-columns: 320px 320px 320px;
    -ms-grid-rows:240px 20px 240px;
    grid-template-rows: 240px 240px;
    grid-gap:20px;
}

/*IE11用 アイテムの設定*/
.container .item:nth-child(1){
    -ms-grid-row: 1;
    -ms-grid-column: 1;
}
.container .item:nth-child(2){
    -ms-grid-row: 1;
    -ms-grid-column: 3;
}
```

CSS Gridを有効化

列（column）の幅と数を指定

行（row）の高さと数を指定

アイテム同士の間隔（縦も横も）

1行目、1列目

1行目、3列目

memo

IE11にはアイテム同士の間隔を指定するgrid-gapが効かないため、隙間部分もグリッドの箱として配置する必要があります。IE11用の記述（-ms-grid-columnsと-ms-grid-rows）の値に、1つおきにgrid-gapと同じ数値（ここでは20px）を記述します。

memo

偶数行目、偶数列目はIE11用の隙間を作るための箱になるので、奇数行目、奇数列目に要素を配置。

↓次ページへ続く

CSS ↓前ページからの続き

```
.container .item:nth-child(3){
    -ms-grid-row: 1;
    -ms-grid-column: 5;
}
.container .item:nth-child(4){
    -ms-grid-row: 3;
    -ms-grid-column: 1;
}
.container .item:nth-child(5){
    -ms-grid-row: 3;
    -ms-grid-column: 3;
}
.container .item:nth-child(6){
    -ms-grid-row: 3;
    -ms-grid-column: 5;
}
```

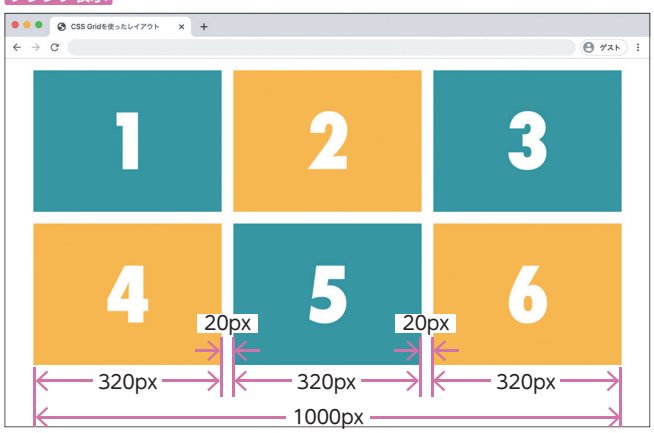
1行目、5列目

3行目、1列目

3行目、3列目

3行目、5列目

ブラウザ表示



レスポンシブにする

ブラウザの幅が1000pxを切るとアイテムがはみ出るので、ブレイクポイントを999pxにします。先ほどの図2では幅320pxの列が3つだったのに対し、図3のCSSでは、155pxの列が2つになっています。代わりに行が増えています。

図3 レスポンシブにする

CSS

```
@media screen and (max-width: 999px){
    .container{
        width: 320px;
        -ms-grid-columns: 155px 10px 155px;
        grid-template-columns: 155px 155px;
        -ms-grid-rows: 116px 10px 116px 10px 116px;
        grid-template-rows: 116px 116px 116px;
        grid-gap:10px;
    }
/*IE11 用 アイテムの設定 */
    .container .item:nth-child(1){
        -ms-grid-row: 1;
        -ms-grid-column: 1;
    }
    .container .item:nth-child(2){
        -ms-grid-row: 1;
        -ms-grid-column: 3;
    }
    .container .item:nth-child(3){
        -ms-grid-row: 3;
        -ms-grid-column: 1;
    }
    .container .item:nth-child(4){
        -ms-grid-row: 3;
        -ms-grid-column: 3;
    }
    .container .item:nth-child(5){
        -ms-grid-row: 5;
        -ms-grid-column: 1;
    }
    .container .item:nth-child(6){
        -ms-grid-row: 5;
        -ms-grid-column: 3;
    }
}
```

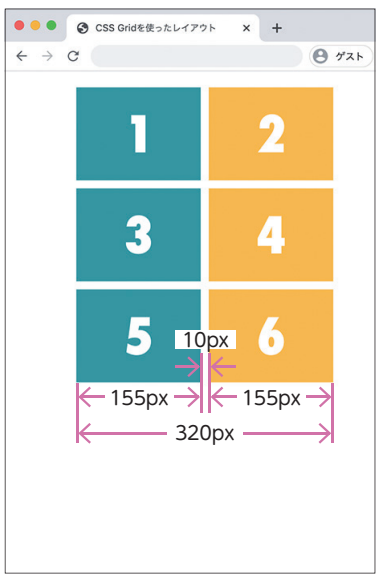
要素がはみださないブレイクポイントを指定

小さいデバイスに合わせた幅

width に収まるよう、2 列 3 行に変更

IE 用にアイテムの位置を再設定

モバイルブラウザ表示



可変幅にしてみよう

図1 ～ 図3 のCSSでは各グリッドの幅やコンテナの幅をpxで指定しているため、計算が面倒です。また、999px以下になった途端余白が大きく生まれてしまいます。CSS Gridには便利な可変の単位（fr）が用意されているため、それを使ってもう少し簡単に、かつ最適表示にできるようにしてみましょう 図4（HTMLは 図1 と同じです）。

> **memo**
> 紙面上では 図2 図3 との違いがわかりにくいため、実際に自分でブラウザで開き、ウィンドウ幅を変えて確認してみましょう。

図4 （frを使った可変バージョンのCSS）

CSS

```
/* 全体の設定・事前準備 */
*{
    box-sizing:border-box;
    margin: 0;
    padding: 0;
}
ul{list-style: none;}
img{
    display: block;
    max-width: 100%;
}
.container{
    width: 100%;
    max-width: 1000px;
    margin: 20px auto;
}
.container{
    display: -ms-grid;
    display: grid;
    -ms-grid-columns: 1fr 20px 1fr 20px 1fr;
    grid-template-columns: 1fr 1fr 1fr;
    -ms-grid-rows: 1fr 20px 1fr;
    grid-template-rows: 1fr 1fr;
    grid-gap:20px;
}

.container .item:nth-child(1){
    -ms-grid-row: 1;
    -ms-grid-column: 1;
}
    （省略）
.container .item:nth-child(6){
    -ms-grid-row: 3;
```

図1 とほぼ同じ

ここだけ変更
※コンテナの幅を100%にし、大きなブラウザで見ても1000pxを超えないようにしている

トラックの幅を 1:1:1 で 3 等分

トラックの高さを 1:1 で 2 等分（コンテナに高さは指定されていないので autoのイメージ）

IE11用 アイテムの設定（ 図2 と同じ）

> **memo**
> 「トラック」とは「グリッドトラック」ともいい、「グリッドロウ」（行）と「グリッドカラム」（列）の総称です。

↓次ページへ続く

```
    -ms-grid-column: 5;
}

@media screen and (max-width: 650px){
    .container{
        -ms-grid-columns: 1fr 10px 1fr;
        grid-template-columns: 1fr 1fr;
        -ms-grid-rows: 1fr 10px 1fr 10px 1fr;
        grid-template-rows: 1fr 1fr 1fr;
        grid-gap:10px;
    }

    .container .item:nth-child(1){
        -ms-grid-row: 1;
        -ms-grid-column: 1;
    }
      （省略）
    .container .item:nth-child(6){
        -ms-grid-row: 5;
        -ms-grid-column: 3;
    }
}
```

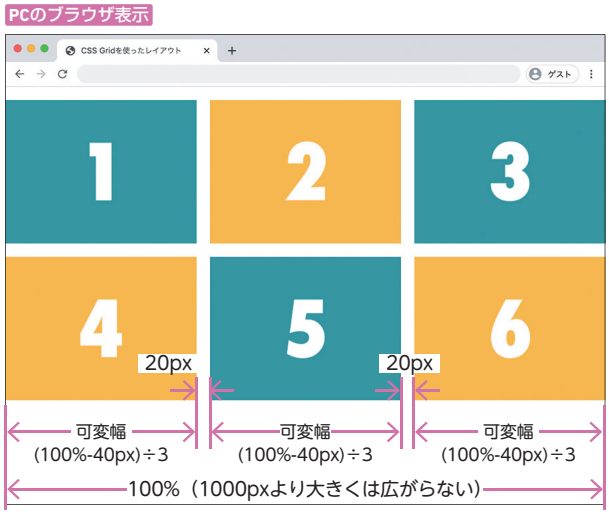
画像の大きさが320pxなので、320+320+10(gap)=650

モバイル向けにもfrを使う
列と行の数を変えただけ

間隔も少し狭く

IE11用 アイテムの設定
（図3と同じ）

PCのブラウザ表示



モバイルのブラウザ表示