three.js の仕様変更に伴う変更点

three.js は現在も数多くの機能追加、改訂などが行われておりますが、その際にクラスやメソッド、プロパティの名称や機能が変更になっている場合もあります。つまり、過去のリビジョンで動作していたものが最新のリビジョンでは動かないということが多々あるという問題点があります。しかしながら、リビジョンが上がるほど使いやすくまたバグも減ってきているので、基本的にはその都度の最新版を利用していくことをおすすめします。本項では、これまでに出版された以下の書籍をリビジョン r70 で動作させるために必要な修正点について記載します（リビジョンごとの変更履歴のなどの詳しい情報は「https://github.com/mrdoob/three.js/」に記載されています)。

【対象書籍】

■HTML5 による物理シミュレーション

■HTML5 による物理シミュレーション【拡散・波動編】

クラス名の変更：THREE.CubeGeometry　→　THREE.BoxGeometry

立方体オブジェクト生成するクラスである CubeGeometry クラスの名称が BoxGeometry クラスへと変更になりました。もともと、３辺の長さを任意に指定することができるので直方体と表すほうが適切なので、この名称変更は妥当です。ただし、利用方法などに変更はありません。

position プロパティへの直接代入不可

position プロパティは Object3D クラスのプロパティで、一般的にはグローバル座標系における３次元グラフィックスの位置座標を Vector3 クラスのオブジェクトで指定します。この position プロパティへ値を与える際に、Vector3 クラスのオブジェクトを直接代入することが無効とされました。例えば、three. js のカメラオブジェクト camera の位置を指定する際に

```
camera.position = new THREE.Vector3(10,10,10);
```

といった記述で値を更新することができません。そのため、Vector3 クラスの clone メソッドを利用しての位置座標の更新も不可能となります。例えば、

```
○○.position = this.r.clone();　//ダメな例（これまでの実装）
```

といった記述では値を更新ができません。対応策として Vector3 クラスの copy メソッドを利用して、位置座標のコピーを行います。具体的には

```
○○.position.copy ( this.r );    //適切な例（これからの実装）
```

というような記述に変更します。

WebGLRenderer クラスのアルファテストのデフォルト値の変更

three.js の WebGLRenderer クラスはレンダリングに必要な情報を含むクラスです。レンダリング時の挙動を指定するプロパティの一つであるアルファテスト適用の有無を指定する alpha プロパティのデフォルト値が変更になりました。

```
renderer = new THREE.WebGLRenderer({
        antialias: true,    //アンチエイリアス（デフォルト：false）
        alpha: true          //アルファテスト（デフォルト：false）←----------追加
});
```

なお、WebGL ではアルファテストのデフォルト値は true です。にも関わらす three.js にて false をデフォルトした理由として考えられるのはパフォーマンスの問題です。

THREE.WebGLRenderer クラスのメソッド名の変更：setClearColorHex → setClearColor

背景色を指定するメソッドの名称が変更になりました。なお、引数の与え方に変更はありません。

THREE.MeshPhongMaterial クラスにおける鏡面色計算方法の変更

フォン反射材質による鏡面色の計算方法が変更されました。その結果、ポリゴン面が描画色が白色に飛んでしまう場合、16 進数形式の specular の値を小さくしてみてください。例：「specular: 0xFFFFFF」→「specular: 0x080808」

クラス実装の変更：THREE.PlaneGeometry → THREE. PlaneBufferGeometry

平面オブジェクト生成するクラスである PlaneGeometry クラスの低メモリ消費版として PlaneBufferGeometry クラスが定義されました。PlaneGeometry クラスを利用してもコンストラクタ内で、PlaneBufferGeometry クラスのコンストラクタが実行されていることから、今後本クラスの名称変更の可能性があります。PlaneGeometry クラスを利用してもコンソール画面にエラーやワーニングが表示されるわけではありませんが、console.info による「THREE.PlaneGeometry: Consider using THREE.PlaneBufferGeometry for lower memory footprint.」というインフォメーションが表示されることから、PlaneBufferGeometry クラスを利用することを推奨します。なお、利用方法などに変更はありません。

クラスの廃止：THREE.Face4 クラス

three.js では、Geometry クラスのオブジェクトに指定した頂点データを利用してポリ

ゴン面を指定する仕組みとして、リビジョン 56 では Face3 クラスと Face4 クラスが存在していました。しかしながら、Face4 クラスによるポリゴン面を指定する実装は、コンストラクタ内で実質的に Face3 クラスを呼び出しているだけなので、本質的には必要ではありませんでした。そのため、リビジョン 60 で廃止されました。Face4 クラスから Face3 クラスへの具体的な変更の具体的方法として、【拡散・波動編】3.1.2 項「２次元格子オブジェクト（2DLattice.html）」を例に上げます。

旧実装（Face4 の利用）

```
for (j = 0; j < N; j++) {
	for (i = 0; i < N; i++) {
		//四角形オブジェクトの頂点インデックスの算出と面オブジェクトの生成
		var face = new THREE.Face4(
			(N + 1) * j + i,		//左下
			(N + 1) * j + i + 1,	//右下
			(N + 1) * (j + 1) + i + 1, //右上
			(N + 1) * (j + 1) + i);	//左上

		//面指定用頂点インデックスを追加
		geometry.faces.push(face);
	}
}
```

新実装（Face3 の利用）

```
//面指定配列の準備
for (var i = 0; i < N; i++) {
	for (var j = 0; j < N; j++) {
		//頂点番号
		var ii = (N + 1) * i + j;
		//面指定用頂点インデックスを追加
		geometry.faces.push( new THREE.Face3( ii, ii + (N + 1), ii + (N + 1) +
1) );
		//面指定用頂点インデックスを追加
		geometry.faces.push( new THREE.Face3( ii, ii + (N + 1) + 1, ii + 1 ) );
	}
}
```