
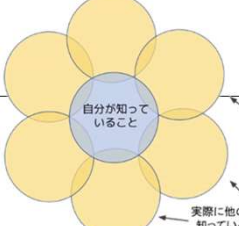


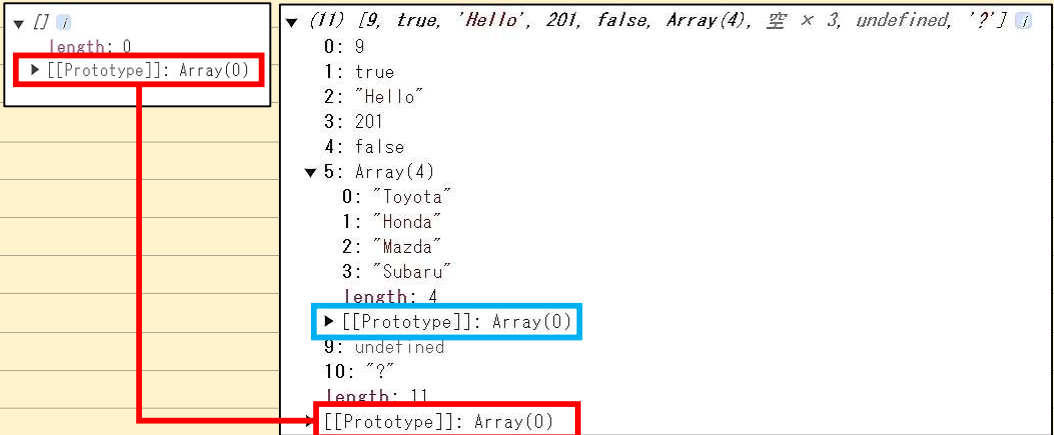
## ■教育実施記録帳

管理No.	FND27	Day7	Page1
実施内容	アセスメント1、インポスターシンドローム、配列入門		
場所	ライオン	設備・工程	
実施日	2024/5/10	実施時間	9:00 ~ 12:00
実施者	Seijiroさん	受講者	
作成者	本田	作成日(開始)	2024/5/10
		作成日(完了)	2024/5/10
<Time>	<Contents>		
	<b>★インポスター症候群とは？</b> <ul style="list-style-type: none"> <li>・クルーが隣に複数いてもキルしたくなる衝動が抑えられない症状の総称...ではありません</li> <li>・やれている客観的証拠が有るのに、自分は能力不足だと思い込む心理状態</li> <li>☆特に才能のある女性に多いことで知られている</li> </ul>		
	<div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="text-align: center;"> <p>インポスター症候群の心理状態</p>  </div> <div style="text-align: center;"> <p>現実</p>  </div> </div>		
	<b>★よくある2つの感情パターン</b>		
	<b>1. 「自分に対する評価は二セモノ」と感じてしまう</b>		
	<ul style="list-style-type: none"> <li>a. 失敗しそうだと思い込んで必死にまたは過剰に働く</li> <li>b. 自分の能力のなさがばれてしまうと強いストレスを感じる</li> </ul>		
	<b>2. 自身の成功を軽視する</b>		
	<ul style="list-style-type: none"> <li>a. 褒められても素直に受け取れない</li> <li>b. 自分自身の成功を認められない</li> <li>c. よく知っていることでも自信がない</li> <li>d. 自分の能力を疑う</li> </ul>		
	<b>★インポスター症候群と向き合うためには</b> <ul style="list-style-type: none"> <li>・これから成長速度ではなく、これまでの道のりを振り返ってみる</li> <li>☆毎週、自分がどれだけのことを達成したかを書き出してみる</li> <li>・助けが必要な時（心が欲している時）に助けを求めるのは良いこと</li> <li>☆知らないことは知らないと言うことに慣れる（だから今から知っていくんだと思えば項垂れることもない）</li> <li>・一歩引いてものごとを俯瞰的に考える（背後に立って自分を見下ろしてみる的なイメージ）</li> </ul>		
	<b>★ARRAY(配列[])とは</b> <ul style="list-style-type: none"> <li>・複数の異なるデータ型の値を1つの変数に保存できる</li> <li>・配列とは複数の値のリスト</li> </ul>		
	<div style="display: flex; justify-content: space-between;"> <div>// 配列の作成</div> <div>リテラル表現</div> </div>		
	<pre>const carBrands = [ "Toyota", "Honda", "Mazda", "Subaru" ];</pre>		
	<pre>    変数名      = [ 要素1 , 要素2 , 要素3 , 要素4  ];</pre>		
	<pre>    インデックス ⇒ [ 0 , 1 , 2 , 3  ];</pre>		
	<pre>const info = [ 9 , true , "Hello", 201 , false , carBrands ];</pre>		
	<pre>    変数名      = [ 要素1 , 要素2 , 要素3 , 要素4 , 要素5 , 要素6  ];</pre>		
	<pre>    インデックス ⇒ [ 0 , 1 , 2 , 3 , 4 , 5  ];</pre>		
	<ul style="list-style-type: none"> <li>・リストの値をそれぞれ<b>要素</b>という</li> <li>・<b>インデックス</b>（添え字）は<b>0</b>から始まる</li> <li>・要素間は、<b>カンマ（コンマ）</b>で仕切る</li> <li>・要素は<b>[]角カッコ</b>で囲われる</li> </ul>		
	☆要素数の限界はメモリ容量との殴り合い、勝てる気がしないからメモリエラーになるまではそっとしとく		

## ■教育実施記録帳

管理No.	FND27	Day7	Page2
実施内容	アセスメント1、インボスターシンドローム、配列入門		
場所	ライオン	設備・工数	
実施日	2024/5/10	実施時間	9:00 ~ 12:00
実施者	Seijiroさん	受講者	
作成者	本田	作成日(開始)	2024/5/10
		作成日(完了)	2024/5/10
<Time>	<Contents>		
	★配列へのアクセス		
	// 配列の作成		
	const carBrands = ["Toyota", "Honda", "Mazda", "Subaru"];		
	const info = [9, true, "Hello", 201, false, carBrands];		
	// 配列へのアクセス		
	console.log( info [ 2 ] ); // ⇒ Hello		
	変数名 [ インデックス ]		
	// 例1		
	console.log( info [5] ); // ⇒ (4) ["Toyota", "Honda", "Mazda", "Subaru"]		
	console.log( carBrands [0] ); // ⇒ Toyota		
	console.log( info [5][0] ); // ⇒ Toyota		
	// 例2		
	console.log( info ); // ⇒ (6) [9, true, "Hello", 201, false, Array(4)]		
	★配列への代入		
	// 配列の作成		
	const carBrands = ["Toyota", "Honda", "Mazda", "Subaru"];		
	const info = [9, true, "Hello", 201, false, carBrands];		
	// 配列への代入		
	info [0] = "nine";		
	console.log( info ); // ⇒ (6) ["nine", true, "Hello", 201, false, Array(4)]		
	// 例1		
	info [3] = true;		
	console.log( info ); // ⇒ (6) [9, true, "Hello", true, false, Array(4)]		
	// 例2		
	info [2 + 2] = 4;		
	console.log( info ); // ⇒ (6) [9, true, "Hello", 201, 4, Array(4)]		
	// 例3		
	info [3 * 2] = "Boo!";		
	console.log( info ); // ⇒ (7) [9, true, "Hello", 201, false, Array(4), "Boo!"]		
	// 疑問1		
	info [5 * 2] = "?";		
	console.log( info ); // ⇒ (11) [9, true, "Hello", 201, false, Array(4), 空 × 4, "?"]		
	・空の要素を作ってしまうと「長さはあるけど要素がない」配列になる。「疎な配列」と呼ぶらしい。		
	// 疑問1の状態では疑問2		
	console.log( typeof info [9] ); // ⇒ undefined		
	// 疑問2の状態では疑問3		
	info [9] = undefined;		
	console.log( info ); // ⇒ (11) [9, true, 'Hello', 201, false, Array(4), 空 × 3, undefined, '?']		
	・空要素はtypeofでundefinedを返したけど実際にはundefinedではない…何を言ってるのやら…		
	・undefinedを与えたら空要素が「僕undefined!」って名乗り始めたからundefinedはいなかったことになる		
	・箱の10個目の仕切りの中にundefinedすらいなくて「undefinedってことにしといて!」っていうメモ書きが外側に貼られてるイメージ…かな?		
	//用途とか問題点とかは今度にしよう…		
	<div> <div>9</div> <div>true</div> <div>"Hello"</div> <div>201</div> <div>false</div> <div>Array</div> <div>unde</div> <div>"?"</div> </div>		
	undefinedってことで!		

## ■教育実施記録帳

管理No.	FND27	Day7	Page3
実施内容	アセスメント1、インボスターシンドローム、配列入門		
場所	ライオン	設備・工数	
実施日	2024/5/10	実施時間	9:00 ~ 12:00
実施者	Seijiroさん	受講者	
作成者	本田	作成日(開始)	2024/5/10
		作成日(完了)	2024/5/10
<Time>	<Contents>		
	★.push()メソッド		メソッド
	<ul style="list-style-type: none"> <li>・配列の最後に要素を付け足す</li> <li>・返値として配列の新しい長さを返す</li> </ul>		
	// 配列の作成		
	const info = [];		//(〆〆)??この状態って疎な配列と同じでは…
	console.log(info); // ⇒ []		//[]は[]…気のせいかな…
			
	<ul style="list-style-type: none"> <li>・配列末尾のArray(0)、空(疎の配列)もArray(0)</li> <li>要するに疎の配列は「長さ(0)だけ与えられた初期化された配列」と同じものってことが判明</li> </ul>		
	// 配列の作成		//(〆〆)で??って感じだけど話を戻そう
	const info = [];		
	// 配列のプッシュ		
	info.push(9);		
	console.log(info); // ⇒ [9]		
	info.push(true);		
	console.log(info); // ⇒ [9, true]		
	info.push("Hello");		
	console.log(info); // ⇒ [9, true, "Hello"]		
	info.push(201);		
	console.log(info); // ⇒ [9, true, "Hello", 201]		//疎の配列にならずに済む
	★.lengthプロパティ		プロパティ
	<ul style="list-style-type: none"> <li>・配列の長さを表す</li> </ul>		
	// 配列の作成		
	const info = [9, true, "Hello", 201];		
	// 配列の要素の数を調べる		
	console.log(info.length); // ⇒ [4]		
	★.pop()メソッド		
	<ul style="list-style-type: none"> <li>・配列から最後の要素を取り除く</li> </ul>		
	// 配列の作成		
	const info = [9, true, "Hello", 201];		
	// 配列から最後の要素を取り除く		
	info.pop();		
	console.log(info); // ⇒ (3) [9, true, "Hello"]		

## ■教育実施記録帳

管理No.	FND27	Day7	Page4			
実施内容	アセスメント1、インボスターシンドローム、配列入門					
場所	ライオン	設備・工数				
実施日	2024/5/10	実施時間	9:00	～	12:00	
実施者	Seijiroさん	受講者				
作成者	本田		作成日(開始)	2024/5/10	作成日(完了)	2024/5/10
<Time>	<Contents>					
	★.unshift()メソッド					
	・配列の最初に要素を付け足す					
	・新しい配列の長さを返す					
	// 配列の作成					
	const info = [9, true, "Hello", 201];					
	// 配列の要素の数を調べる					
	info.unshift("nine");					
	console.log(info); // ⇒ (5) ["nine", 9, true, 'Hello', 201]					
	★.shift()メソッド					
	・配列の最初から要素を削除する					
	・新しい配列の長さを返す					
	// 配列の作成					
	const info = [9, true, "Hello", 201];					
	// 配列の要素の数を調べる					
	info.shift();					
	console.log(info); // ⇒ (3) [true, 'Hello', 201]					
	★.concat()メソッド					
	・二つ以上の配列を結合する					
	// 配列の作成					
	const array1 = ["a", "b", "c"];					
	const array2 = ["d", "e", "f"];					
	// 配列の結合					
	const array3 = array1.concat(array2);					
	console.log(array3); // ⇒ (6) ['a', 'b', 'c', 'd', 'e', 'f']					
	★ 配列宣言時の注意点					
	・配列宣言はconstで行うこと					
	☆letで行うと意図せず再代入の危険がある					
	<div>Masa@DIG から 全員 11:21</div> <div>「再代入できない」ではなくて「再代入を防いでくれる」といった振る舞いですね</div>					
	★ミュータブルデータ					
	・作成後にその中身を変えることのできるデータ					
	・配列はミュータブルデータ					
	★イミュータブルデータ					
	・作成後にその中身を変えることのできないデータ					
	・数値、文字列、boolean、undefinedはイミュータブルデータ					
	★JSON.stringify()メソッド					
	・二つの配列が等しいことの確認をするときに使用する					
	・値をJSON文字列に置き換えて比較する					
	//配列の比較					
	const array1 = ["a", "b", "c"];					
	const array2 = ["d", "e", "f"];					
	JSON.stringify(array1) === JSON.stringify(array2)					

JSON