


■教育実施記録帳

管理No.	FND27	Day13	Page1
実施内容	クロージャ		
場所	ライオン	設備・工数	
実施日	2024/5/20	実施時間	9:00 ~ 12:00
実施者	Masaさん	受講者	
作成者	本田	作成日(開始)	2024/5/20
		作成日(完了)	2024/5/20
<Time>	<Contents>		
	★クロージャ ・レキシカルスコープの変数を関数が使用している状態のこと		
			
	★レキシカルスコープ ・実行中のコードから見た外部スコープのこと		
	[. js] レキシカルスコープ		
	<pre>let a = 1; function fn1() { let b = 2; function fn2() { let c = 3; } fn2(); } fn1();</pre>		
			
	★スコープチェーン ・関数内に変数が見つからない場合、外側のスコープに変数を探しに行く仕組み		
	[. js] スコープチェーン		
	<pre>let a = 1; function fn1() { let a = 2; function fn2() { console.log(a); // 関数内に変数aが定義されていないため、外側のスコープに探しに行く } fn2(); } fn1();</pre>		
	[DevTool Console]		
	2		

■教育実施記録帳

管理No.	FND27	Day13	Page2
実施内容	クロージャ		
場所	ライオン	設備・工機	
実施日	2024/5/20	実施時間	9:00 ~ 12:00
実施者	Masaさん	受講者	
作成者	本田	作成日(開始)	2024/5/20
		作成日(完了)	2024/5/20
<Time>	<Contents>		
	★クロージャの活用方法1		
	[. js] データのプライバシー保護		
	<pre>function outer() { let count = 1; function counter() { console.log(count); count += 1; } return counter; }</pre>		
	//関数スコープの中に全体を入れる		
	//counterはグローバルスコープで使うためreturnで返す		
			
	count = 5; //グローバルスコープからアクセス出来ず、count変数は保護される		
	[DevTool Console]		
	ReferenceError: count is not defined		
	[. js] クロージャのおかげ		
	<pre>function outerFunc() { let count = 1; function counter() { console.log(count); count += 1; } return counter; }</pre>		
	//outerFuncを実行する (②)		
	//クロージャ		
	outerFuncの処理が終わっても、内側の関数が外側の変数countを参照しているため使い続けられる		
	//outerFunc()の返回值を返す (③)		
	const counter1 = outerFunc(); //counter〇〇を作るときにcountが組み込まれる 関数式		
	↑ クロージャのおかげ		
	counter1(); ① //counter1を実行する (①)		
	counter1(); ②		
	[DevTool Console]		
	1		
	2		

■教育実施記録帳

管理No.	FND27	Day13	Page3
実施内容	クロージャ		
場所	オンライン	設備・工数	
実施日	2024/5/20	実施時間	9:00 ~ 12:00
実施者	Masaさん	受講者	
作成者	本田	作成日(開始)	2024/5/20
		作成日(完了)	2024/5/20
<Time>	<Contents>		
	★クロージャの活用方法2		
	[. js] 同じような関数をたくさん作る場合にただ延々書くのは大変		
	function taxCalTokyo(price) {		
	return price * 0.1;		
	}		
	function taxCalNY(price) {		
	return price * 0.08875;		
	}		
	function taxCalParis(price) { //何十都市となると、書くのも見直すのも大変		
	return price * 0.2;		
	}		
	[. js] 関数シグネチャを整理する		
	taxCal //税額を計算する関数		
	引数 : price		
	戻り値 : priceに税率を掛けたもの		
	taxCalMaker //taxCalを作る		
	引数 : 税率		
	戻り値 : taxCal		
	[. js] 同じような関数をまとめてたくさん作る		
	function taxCalMaker(taxRate) {		
	function taxCal(price) {		
	return price * taxRate;		
	}		
	return taxCal;		
	}		
	const taxCalTokyo = taxCalMaker(0.1);		
	const taxCalNY = taxCalMaker(0.08875);		
	const taxCalParis = taxCalMaker(0.2);		
	taxCalTokyo(1000);		
	taxCalNY(1000);		
	taxCalParis(1000);		
	[DevTool Console]		
	100		
	88.75		
	200		