プログメントのソースに関するドキュメント

松本拓也

平成 27 年 1 月 29 日

1 programingInitializeModule

1.1 グローバル変数

表 1: グローバル変数一覧

変数名	データ型	初期値	備考
codeStr	String	空文字	実行コードの代入先
preamble	String	空文字	実行コードの変数宣言などの処理部
subroutine	String	空文字	実行コードのサブルーチン部
varNum	int	0	未命名変数の命名用カウント変数
varNameTable	ObjectArray	{hoge:”foo”}	変数名管理用連想配列
funNum	int	0	未命名サブルーチン命名用のカウント変数
functionNameTable	ObjectArray	{hoge:”foo”}	サブルーチン名管理用連想配列
exeMode	String	”nomal”	実行モード指定用変数
hogeNum	int	0	ループ時に使用する変数のカウント変数
blockNum	int	0	ブロックに ID を振る場合に使用するカウント変数
funlib	String	空文字	ライブラリ読み込み用変数
stackNum	int	0	トレース実行時のスタックポインタ
generator	Array	空	トレース実行でイテレータに使うジェネレータ
result	Object	undefined	トレース実行の処理結果を格納する変数
OutText	String	空文字	トレース実行時の出力処理に用いる変数
programingMode	String	$(”#programmingMode”).val()	プログラミングモードの識別用変数

1.2 reInitialize

プログラムを新たに読み込んだ際に workspace 中の属性を再付与する必要が有るため，読み込み時に再初期化を行うためのメソッド．基本的には Ready 内で行っている処理と同一．

1.3 initializeBlockList

ブロックリスト内のオリジナルとなるブロックの初期化処理．付与しているのは Draggable によるドラッグ属性で helper に clone を使用している．ネスト構造やインプットタグを含んでいるものにはその要素をキャンセルするオプションを付与している．また，contaiment オプションを用いて workspace 内のみをドラッグするようにしている．zIndex と revert については各種適宜数値を設定している．

1.4 getBlockType

引数にオブジェクト（ブロックのオブジェクトを受け取るはず）が渡され，そのオブジェクトがなんのブロックであるかを判断して文字列で結果を返すメソッド．jQueryUi により，クラスが追加されているため，ブロックの状態によってブロックに割り当てられているクラス数が変動する．そのため一番後ろから順に解析することでブロックの種類に当たるクラスを検出して結果を返す処理を行う．該当クラスが最後まで見つからなかった場合は”unknown”を返す．

1.5 initializeWorkspace

workspace 内のブロックに属性の付加を行う初期化メソッド

1.6 annexDraggable

引数として受け取ったブロックのオブジェクトに種類に応じて draggable 処理を行う．新たに生成したブロックや workspace の初期化処理で利用する．

1.7 initializeKadaiMode

課題モード時に必要な初期化処理を行うメソッド．ブロック制限情報を格納するグローバル変数 limitList を宣言して処理を行う．

1.8 forbiddenDraggingWorkspace

課題モード時に課題ルーチン以外の workspace 中のブロックを操作できないようにドラッグ機能を停止するメソッド．

1.9 blockGenerateControl

内部設計書に記述していたブロックリスト管理モジュールをプログラミング初期化モジュールに内包させている．

ブロックリストの制限を行うための初期化処理を行うメソッド．一度ブロックリストの全ブロックをドラッグ禁止及び非表示にしたあと，ブロック制限リストで仕様が許可されているブロックのみドラッグの許可と表示を行う．

1.10 blockGenerated

課題モード時にブロックを生成した際に呼び出されるメソッド．引数として受け取ったブロックのオブジェクトからブロックの種類を識別し，そのブロックの使用可能個数をデクリメントする．使用可能数が 0 になる場合はそのブロックのオリジナルを非表示とし，ドラッグ禁止にする．

1.11 blockDeleted

課題モード時にブロックを削除した際に呼び出されるメソッド．引数として受け取ったブロックのオブジェクトからブロックの種類を識別し，そのブロックの使用可能個数をインクリメントする．生成時とは異なり，削除するブロックが内部に別のブロックを持っている可能性があるため，子要素を調べて再帰的に処理することで走査している．使用可能数が 0 から 1 以上になる場合はそのブロックのオリジナルを再表示し，ドラッグを許可する．

2 executionModule

実行系のメソッドが記述されている．

2.1 ready 内

2.1.1 exeButton クリック時の動作

解析・実行処理を行うメイン関数を呼び出して処理を行う．その後トレースモードであればトレース実行のボタンに表示をスイッチする．課題モードで通常実行を行った際は正誤判定を行う．正解の際は正解という文字を sweetAlert を用いて表示する．不正解の場合は正解の場合の出力結果を表示することでヒントを与えている．

2.1.2 nextButton クリック次の動作

トレース実行の際に実行ボタンと表示をスイッチして出現するボタンについての処理．最初に result 変数からトレース実行の処理状態を判定する．開始したところの場合はコンソールにメッセージを出力している．終了している場合，もしくは別のモードで実行されてしまった場合は処理を行わずに処理する．

状態の判定後，generator と statckNum から実行中のメソッドを呼び出し，イテレータを用いてステップ実行を行う．その後終了していなければ yield の返り値を出力し処理を終える．処理が終了している場合はその旨を出力エリアに表示しトレース実行に用いていた ID を削除，next ボタンを再度非表示にして実行ボタンを表示する．

2.2 outputTextArea

引数として渡された文字列を出力エリアであるテキストエリアに表示するメソッド．システム内とユーザが作成したプログラム内で利用される．

2.3 traceoutputTextArea

トレース実行時に使用する出力用メソッド．通常の出力メソッドをイテレータを用いた処理で呼び出すことが出来なかったため別途用意したメソッド．基本的には処理内容は同一であるがトレース実行用メッセージを追加で出力している．

2.4 executionMain

実行・解析処理を行う際に利用されるメインメソッド．処理に用いる変数の初期化処理を行った後，サブルーチンの解析を行い，メインルーチンの解析を行う．これらにより得られた解析結果を連結し実行コードを生成する．その後，そのコードをコード生成モード以外では実行，コード生成モードでは表示する．

2.5 functionInterpret

サブルーチンを解析するためのメソッド．workspace 内のサブルーチン全てに対し each を用いて処理を行う．

最初にサブルーチン名の名前の検査を行う．名前が付けられていない場合は名前が被らないように適当な名前を設定する．その後各に対し，関数名の ID を付与する．この際，名前が重複している場合は名前の末尾に番号を付けることで重複を回避している．

メソッド名の検査後，メインルーチンの解析と同様に interpret メソッドを利用してサブルーチンの解析を行い，結果を格納する．トレース実行の場合はサブルーチン呼び出し元に戻るための処理を追記している．

2.6 interpret

処理ブロックや制御構造ブロックを解析するためのメソッド．引数として受け取ったオブジェクトからブロックの種類を getBlockType を用いて調べ，その結果に応じて各種コード生成メソッドを呼び出し，その結果を呼び出し元に返す．トレース実行の際は実行中のブロックに対して付加するエフェクト用クラスを一時的に付与する処理を追記している．

2.7 dataBlockInterpret

データ型のブロックの解析用メソッド．基本的にはこのメソッドでは処理を行わず，データ型に応じて各種メソッドに処理を受け渡す．データブロックが見つからない場合は空の文字列を返す．

2.8 intBlockInterpret

数値型のブロックを処理，解析するメソッド．

2.9 charBlockInterpret

文字列型のブロックを処理，解析するメソッド．

2.10 boolBlockInterpret

論理型のブロックを処理，解析するメソッド．

2.11 各種処理解析

引数として受け取ったブロックのオペランドを解析してそれを元に処理コードを生成して返す．

2.12 getVarName

変数の名前を取得して呼び出し元に返すメソッド．型ごとに内部処理では変数名に加えて型の文字列を付加した変数名に置き換えている．変数名が設定されていない場合は適当な変数名を設定している．また，この処理で調べた変数名が変数名管理テーブルに登録されていない場合は登録処理を行った後，preamble に変数の初期化処理の文を追加する．

2.13 getWorkspace

workspace の保存時に保存用の処理を行った後結果を返す処理．html メソッドではインプットタグの情報が保持されないため html メソッドで取得する前にインプットタグを全域走査して値を取得，置換を行った後セミコロンを置換し結果を返している．

2.14 getResult

課題解析モジュールのメソッド．通常実行を行った後，出力結果を呼び出し元に返す．

3 blockControllerModule

3.1 annexController

引数として受け取ったブロックのオブジェクトからそのブロックが持つドロップ属性やネスト属性を適応する要素を含んでいるかを判定し，その属性を付与するメソッド．実際に付与する処理は各種メソッドで行う．

3.2 annexSort

引数として受け取ったブロックのオブジェクトから子要素の nestArea にソート属性を付加する．ソートしているリストに更新があった場合，その要素がリストからドロップされたものである場合，各種属性の処理を付加する．

3.3 annexIntArea,annexCharArea,annexDataArea,annexBoolArea,varArea

引数として受け取ったブロックの子要素のドロップエリアに各種データ型のドロップ許可の設定などのオプションを設定してドロップ属性を付加する．受け入れ設定が異なるだけで基本的には同一の処理を行う．ドロップエリア内に既にデータが有る場合は要素の追加は行わず処理を終了する．リストから来た場合はクローンを生成した後，各種属性を付加して追加する．それ以外の場合はドロップした要素の CSS 設定を変更した後追加する．