

言語仕様比較

カテゴリ	言語仕様	Curl	ActionScript3.0	Java	備考
変数宣言とデータ型	変数宣言	let str:String = "abc"	var str:String = "abc";	String str = "abc";	
	変数代入	set str = "xyz"	str = "xyz";	str = "xyz";	
	オブジェクト初期化	let monkey:Monkey = (Monkey) or let monkey:Monkey = (new (Monkey))	var monkey:Monkey = new Monkey();	Monkey monkey = new Monkey();	
	型定義なし	let monkey:any	var monkey:* var monkey;	N/A	
	定数修飾子	constant (def)	const	final	
	型チェック演算子	isa type-switch type-of	is type-of	instanceof	
	キャスト	let monkey:Monkey = object as Monkey	var monkey:Monkey = object as Monkey; or var monkey:Monkey = Monkey(object);	Monkey monkey = (Monkey)object;	
	プリミティブ型	int/uint int8/uint8/byte int16/uint16 int64/uint64 char float double bool	int/uint double Boolean	int byte short long char float double boolean	
	日付型	Date, Time	Date	Date, Time, Calendar	
	文字列	String	String	String	
	配列/リスト	let array:(FastArray-of int) = {(FastArray-of int) max-size = 4} let array:(FastArray-of int) = {(FastArray-of int) max-size = 4, 1, 2, 3, 4}  let array:(Array-of int) = {(Array-of int)} let array:(Array-of int) = {(Array-of int) 1, 2, 3, 4}	var array:Array = new Array(); var array:Array = [1, 2, 3, 4];	int[] array = new int[4]; int[] array = {1, 2, 3, 4}  List<Integer> array = new ListArray<Integer>(); array.add(1); array.add(2); array.add(3); array.add(4);	
	連想配列	let h:(HashTable-of String, int) = {(HashTable-of String, int)} {h.set "a", 1} {h.set "b", 2} set h["c"] = 3	var array:Array = new Array(); array["a"] = 1; array["b"] = 2; array["c"] = 3;	Map h = new HashMap(); h.put("a", 1); h.put("b", 2); h.put("c", 3);	
	数量・単位	meter/m degree second/s gram Hz giga ...など	N/A	N/A	
NULL値の許容	宣言時#を付与 let str:String = null Not Nullチェックはif-non-null文	var str:String = null;	String str = null;		
演算子	論理演算子	and or not	&&    !	&&    !	
	足し算	+	+	+	
	引き算	-	-	-	
	掛け算	*	*	*	
	割り算	/	/	/	
	剰余	mod	%	%	
	文字列連結	&	+	+	
	同等	==	== or === (厳密な等価)	==	
	不等	!=	!= or !== (厳密な不等価)	!=	
	小なり	<	<	<	
	大なり	>	>	>	
	以上	<=	<=	<=	
	以下	>=	>=	>=	
値の増減	{inc i} {dec i}	++; --;	++; --;		

言語仕様比較

カテゴリ	言語仕様	Curl	ActionScript3.0	Java	備考
制御文	for文	<pre>{for 条件式 do ... }</pre>	<pre>for (条件式) { ... }</pre>	<pre>for (条件式) { ... }</pre>	
	for ~ in文	・リスト let array:(Array-of int) = ((Array-of int) 1, 2, 3) {for v in arrays do {output v} } ・連想配列 let map:(HashTable-of String, String) = ... {for v key k in map do {output "key=" & k & " value=" & v} }	・リスト/連想配列 var arrays:Array = ... for (var i in arrays) { trace(arrays[i]); }	・リスト List<Integer> array = ... for (Integer a : array) { System.out.println(a); } ・連想配列 Map<String, String> map = ... for (Iterator i=map.keySet().iterator(); i.hasNext()) { String k = (String)i.next(); System.out.println(k + " " + map.get( k)); }	
	while文	<pre>{while 条件式 do ... } or {until 条件式 do ... }</pre>	<pre>while (条件式) { ... } or do { ... } while(条件式);</pre>	<pre>while (条件式) { ... } or do { ... } while(条件式);</pre>	
	if文	<pre>{if 条件式 then ... } elseif 条件式 then ... else ... } {unless 条件式 do ... } {if-non-null 値 then ... }</pre>	<pre>if (条件式) { ... } else if () { ... } else { ... }</pre>	<pre>if (条件式) { ... } else if () { ... } else { ... }</pre>	
	switch文	<pre>{switch 値 case 要素1 do ... case 要素2 do ... else ... }</pre>	<pre>switch (値) { case 要素1 : ... break; case 要素2 : ... break; default : ... }</pre>	<pre>switch (値) { case 要素1 : ... break; case 要素2 : ... break; default : ... }</pre>	
パッケージ/クラス/メソッド	パッケージ宣言	<pre>{package com.samples ... }</pre>	<pre>package com.samples { ... }</pre>	<pre>package com.samples;</pre>	
	パッケージインポート	<pre>import * from com.samples; import Foo from com.samples;</pre>	<pre>import com.samples.*; import com.samples.Foo;</pre>	<pre>import com.samples.*; import com.samples.Foo;</pre>	
	アクセスレベル	<pre>public private package</pre>	<pre>public private protected</pre>	<pre>public private protected</pre>	
	継承	<pre>{define-class public Monkey (inherits Animal) ... }</pre>	<pre>public class Monkey extends Animal { ... }</pre>	<pre>public class Monkey extends Animal { ... }</pre>	
	多重継承	サポート	N/A	N/A	
	プライベートクラス	N/A	N/A	<pre>private class Foo {...}</pre>	
	インターフェース	N/A	<pre>class FooImpl implements Foo {...}</pre>	<pre>class FooImpl implements Foo {...}</pre>	Curlではインターフェースを作れない代わりに多重継承が可能となっている。
	抽象クラス	サポート	N/A	サポート	
	メソッドオーバーライド	サポート	サポート	サポート	
	メソッドオーバーロード	N/A	N/A	サポート	Curlには、"キーワード引数"という概念があり、呼び出し引数を変更可能。
関数クロージャー	サポート	サポート	N/A		
その他	コンソール出力	<pre>{output str} {dump str}</pre>	<pre>trace(str); デバッグモードのみ</pre>	<pre>System.out.println(str);</pre>	
	例外処理	<pre>{try ... catch 例外 do ... finally ... }</pre>	<pre>try { ... } catch (例外) { ... } finally { ... }</pre>	<pre>try { ... } catch (例外) { ... } finally { ... }</pre>	
	コメント	<pre>## 複数行コメント #     単一行コメント</pre>	<pre>/* 複数行コメント */ // 単一行コメント</pre>	<pre>/* 複数行コメント */ // 単一行コメント</pre>	
	パッケージング	<pre>.pcurl</pre>	<pre>.swc</pre>	<pre>.jar</pre>	