

ソフトウェア工学特論 (DSL)

Pokemon.js

杉田基樹

今ポケモンがアツい

ハロワ

アルセウス

ペラップ

アンノーン

ユニラン ビクティニ ビクティニ ヒトカゲ カメール

ユニラン ビクティニ ビクティニ リザードン リザード

ユニラン ビクティニ ビクティニ リザードン バタフリー

ユニラン ビクティニ ビクティニ リザードン バタフリー

ユニラン ビクティニ ビクティニ リザードン スピアー

ユニラン ビクティニ ビクティニ フシギソウ ビクティニ

ユニラン ビクティニ ビクティニ リザードン ゼニガメ

ユニラン ビクティニ ビクティニ リザードン スピアー

ユニラン ビクティニ ビクティニ ゼニガメ フシギソウ

ユニラン ビクティニ ビクティニ リザードン バタフリー

ユニラン ビクティニ ビクティニ リザードン ヒトカゲ

ユニラン ビクティニ ビクティニ フシギソウ フシギダネ

ポケットモンスター・ポケモン・Pokémonは任天堂・クリーチャーズ・ゲームフリークの登録商標です。

目次

1. 概要
2. 言語仕様
 1. プログラム
 2. リテラル
 3. 演算
 4. 出力
 5. 変数
 6. 条件分岐
 7. 反復
3. FizzBuzz
4. まとめ・展望

おことわり

- 本スライドではポケモンの画像を多用します
- 特に記載がない限り, すべて以下のサイトから引用します
 - ポケモンずかん<<https://zukan.pokemon.co.jp/>>
- 商用目的での使用はありません

概要

- Pokemon.jsとは
- 基本仕様
- 工夫した点・苦労した点

1. 概要
2. 言語仕様
 - プログラム
 - リテラル
 - 演算
 - 出力
 - 変数
 - 条件分岐
 - 反復
3. FizzBuzz
4. まとめ・展望

Pokemon.jsとは

- ポケモンの名前のみ使用可能なプログラミング言語
- わずかなポケモンの知識があればすぐに記法を習得できる

ウリ

- わずかなポケモンの知識があればすぐに記法を習得できる
- こどもでも親しみやすい(?)

欠点

- ポケモンの知識が皆無である場合、記法の習得が絶望的に困難
- 全体的に記法が冗長

基本仕様

- ポケモンの名前は次のいずれかで表記できる
 - 日本語名(カタカナ)
 - 英語名(先頭の文字が大文字または小文字の2パターン)
 - 図鑑番号(4桁)
- 空白改行は完全に無視する

工夫した点・苦労した点

- ただのJSのエイリアスにならないように心掛けた
- 使用するポケモンにこだわった
 - ポケモンやっている人なら意図を理解していただけるレベル
- TSで実装した

言語仕様 プログラム

- プログラムの構成
- アルセウス

1. 概要
2. 言語仕様
 - プログラム
 - リテラル
 - 演算
 - 出力
 - 変数
 - 条件分岐
 - 反復
3. FizzBuzz
4. まとめ・展望

プログラムの構成

- はじめに**アルセウス**を記述する
- それ以降, 任意のポケモンを記述可能

アルセウス

...

...

'use strict';

...


...

アルセウス


- ポケモンの世界を
創造したとされるポケモン



分類：そうぞうポケモン

タイプ：
ノーマル

高さ：3.2m 重さ：320.0kg

特性：マルチタイプ  特性

HP 

こうげき 

ぼうぎょ 

とくこう 

とくぼう 

すばやさ 

うちゅうが まだ ない ころに さいしょに うまれた ポケモンと しんわの なかで かた
られている。 (『ポケットモンスター シャイニングパール』より)



言語仕様 リテラル

- 数値
- 文字列
- アンノーン
- 真偽値

1. 概要
2. 言語仕様
 - プログラム
 - リテラル
 - 演算
 - 出力
 - 変数
 - 条件分岐
 - 反復
3. FizzBuzz
4. まとめ・展望

数値

- 図鑑番号0～9のポケモンで10進数を表す



0
ビクティニ



1
フシギダネ



2
フシギソウ



3
フシギバナ



4
ヒトカゲ



5
リザード



6
リザードン



7
ゼニガメ



8
カメール



9
カメックス

数値



- **ドンメル** (英語名: Numel) で数値型を示す
- 後ろに10進数ポケモンをN匹記述し, N桁の整数を表現する

ドンメル
リザードン フシギダネ カメール

618;

```
{  
  "type": "ExpressionStatement",  
  "expression": {  
    "type": "Literal",  
    "value": 618,  
    "raw": "618"  
  }  
}
```

文字列

- 図鑑番号0～15のポケモンで16進数を表す



0



1



2



3



4



5



6



7



8



9



A



B



C



D



E



F

キャタピー トランセル バタフリー ビードル コクーン スピアー

文字列



- アンノーンで文字列, ユニランでUnicodeを示す
- ユニラン + 16進数ポケモン4匹のまとまりをN個記述し, 1つの文字列を表現する

アンノーン

ユニラン リザードン ゼニガメ ヒトカゲ カメックス
ユニラン ゼニガメ リザード フシギバナ ビクティニ
ユニラン フシギソウ リザードン リザードン キャタピー

'杉田¥u266A';

```
{  
  "type": "ExpressionStatement",  
  "expression": {  
    "type": "Literal",  
    "value": "杉田 ♪",  
    "raw": "¥"杉田 ♪¥"  
  }  
}
```

アンノーン

- アルファベット26種・！・？に対応したフォルムがあるポケモン



<https://www.pokemon.jp/special/forme/zukan04/zukan4-1.html>

真理値

- ミュウがTrue, ミュウツーがFalseを表す



True



False

言語仕様 演算

- 算術演算
- 比較演算

1. 概要
2. 言語仕様
 - プログラム
 - リテラル
 - 演算
 - 出力
 - 変数
 - 条件分岐
 - 反復
3. FizzBuzz
4. まとめ・展望

算術演算

- 以下のポケモンが対応



+
プラスル



-
マイナン



*
メタグロス



/
エルレイド



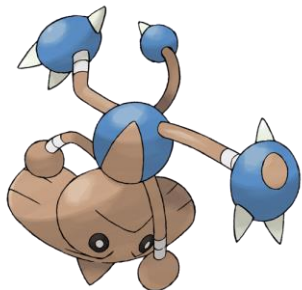
%
アマルルガ

比較演算

- 以下のポケモンが対応



!==
バルキー



===
カポエラー



>
エビワラー



<
サワムラー



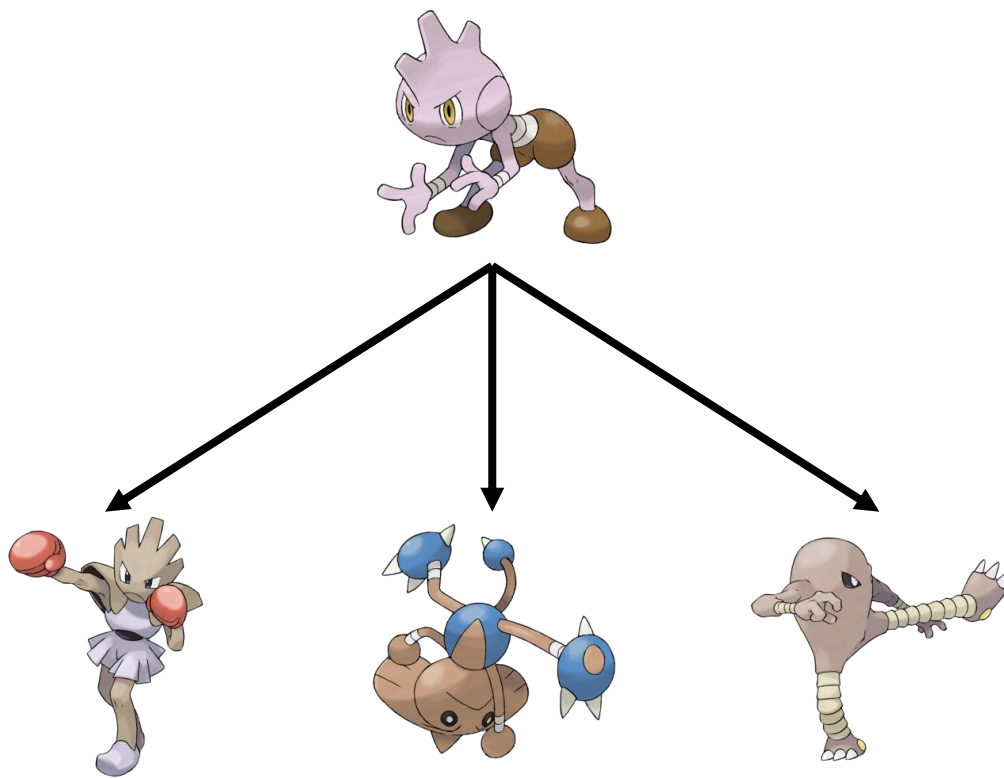
>=
エビワラー
ヤジロン



<=
サワムラー
ヤジロン

比較演算

- 選定理由: 進化のフローに条件分岐が絡んでいるため



バルキー

```
if (防御 > 攻撃) {  
    エビワラーに進化;  
} else if (防御 < 攻撃) {  
    サウムラーに進化;  
} else { // 同じ場合  
    カポエラーに進化  
}
```

言語仕様 出力

- コンソール出力

1. 概要
2. 言語仕様
 - プログラム
 - リテラル
 - 演算
 - 出力
 - 変数
 - 条件分岐
 - 反復
3. FizzBuzz
4. まとめ・展望

コンソール出力



- ペラップの後に式を記述

ペラップ

ドンメル フシギバナ リザードン カメックス
マイナン

ドンメル フシギダネ ヒトカゲ ゼニガメ

```
console.log(369 - 147);
```

```
{
  "type": "ExpressionStatement",
  "expression": {
    "type": "CallExpression",
    "callee": {
      "type": "MemberExpression",
      "computed": false,
      "object": {
        "type": "Identifier",
        "name": "console"
      },
      "property": {
        "type": "Identifier",
        "name": "log"
      }
    },
    "arguments": [
      {
        "type": "BinaryExpression",
        "operator": "-",
        "left": {
          "type": "Literal",
          "value": 369,
          "raw": "369"
        },
        "right": {
          "type": "Literal",
          "value": 147,
          "raw": "147"
        }
      }
    ]
  }
}
```

言語仕様 変数

- 変数宣言
- 識別子
- 代入
- 例

1. 概要
2. 言語仕様
 - プログラム
 - リテラル
 - 演算
 - 出力
 - 変数
 - 条件分岐
 - 反復
3. FizzBuzz
4. まとめ・展望

変数宣言

- メタモン + 識別子 + 式
- `const`にする場合は
クレツフィを記述

`let`



変数
メタモン

?

識別子



式

`const`



変数
メタモン



`const`

?

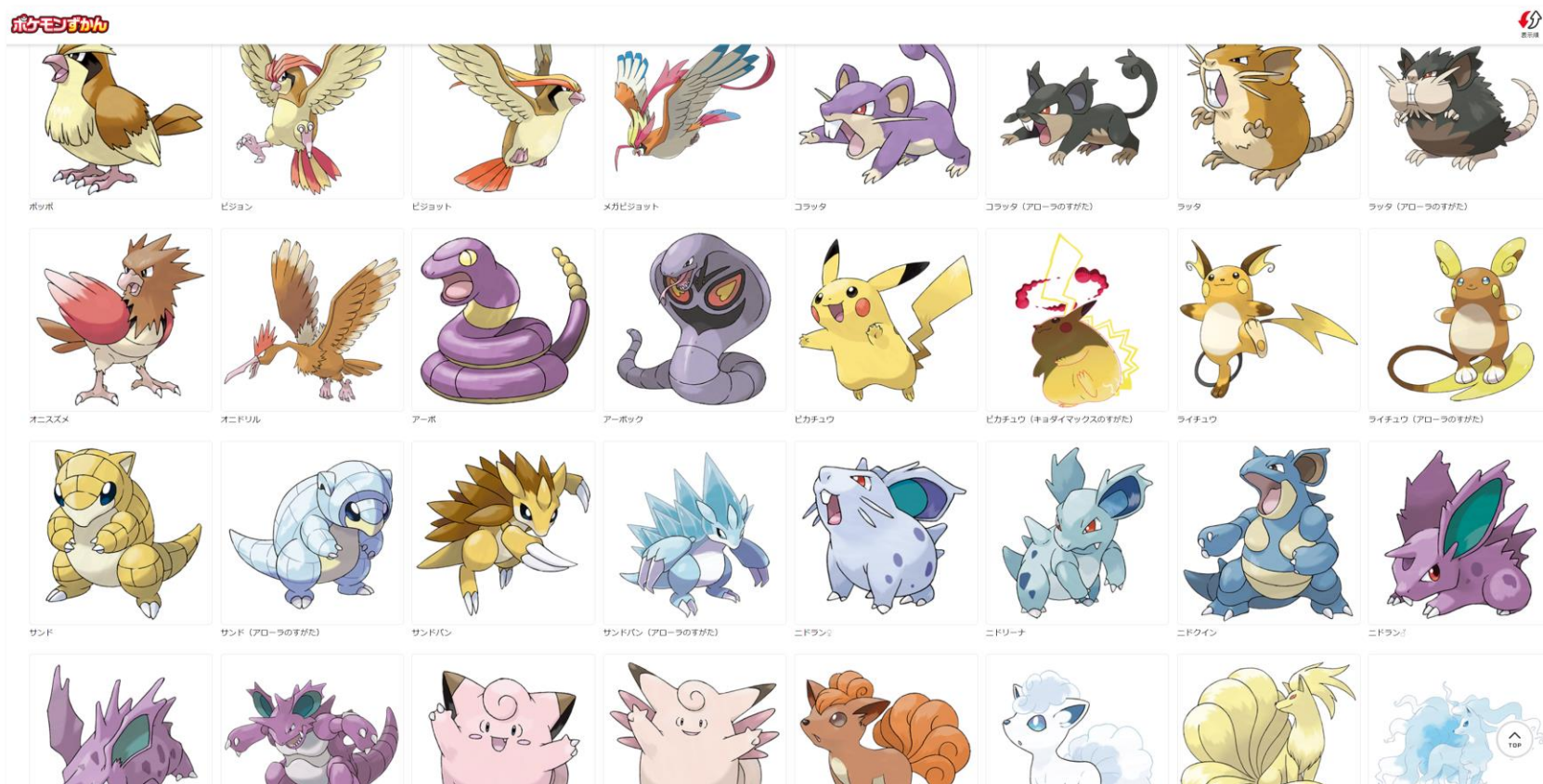
識別子



式

識別子

- 予約語以外すべてのポケモン(約900匹)



代入

- ロトム + 識別子 + 式



代入
ロトム

?

識別子



式

例

メタモン クレツフィ ウインディ
ドンメル フシギソウ
メタモン ニャース
ドンメル フシギダネ ビクティニ
マイナン
ウインディ

ペラップ ニャース
ロトム ニャース ウインディ
ペラップ ニャース



ニャース



ウインディ

```
const arcanine = 2;  
let meowth = 10 - arcanine;  
console.log(meowth);  
meowth = arcanine;  
console.log(meowth);
```

8
2

言語仕様 条件分岐

- 条件分岐記法

1. 概要
2. 言語仕様
 - プログラム
 - リテラル
 - 演算
 - 出力
 - 変数
 - 条件分岐
 - 反復
3. FizzBuzz
4. まとめ・展望

記法

- ヤドン → if
ヤドラン → else if
ヤドキング → else
- 括弧にあたるものを一切使用しない
- ifの終了を表すためにピカチュウを用いる



ヤドン
ヤドラン 条件1
処理1
ヤドラン 条件2
処理2
ヤドラン 条件3
処理3
ヤドキング
処理4
ピカチュウ

言語仕様 反復

- 反復記法

1. 概要
2. 言語仕様
 - プログラム
 - リテラル
 - 演算
 - 出力
 - 変数
 - 条件分岐
 - 反復
3. FizzBuzz
4. まとめ・展望

記法

- セレビィ + 反復条件 + 処理
- 括弧にあたるものを一切使用しない
- whileの終了を表すためにピカチュウを用いる

セレビィ
反復条件
処理
ピカチュウ



FizzBuzz

- ソースコード
- JS・実行結果

1. 概要
2. 言語仕様
 - プログラム
 - リテラル
 - 演算
 - 出力
 - 変数
 - 条件分岐
 - 反復
3. FizzBuzz
4. まとめ・展望

アルセウス

メタモン ポップォ ドンメル フシギダネ

セレビィ

ポップォ サワムラー ヤジロン ドンメル フシギダネ リザード

ヤドン

ヤドラン

ポップォ アマルルガ ドンメル フシギバナ カポエラー ドンメル ビクティニ

ヤドン

ヤドラン

ポップォ アマルルガ ドンメル リザード カポエラー ドンメル ビクティニ

ペラップ

アンノーン

ユニラン ビクティニ ビクティニ ヒトカゲ リザードン
ユニラン ビクティニ ビクティニ リザードン カメックス
ユニラン ビクティニ ビクティニ ゼニガメ キャタピー
ユニラン ビクティニ ビクティニ ゼニガメ キャタピー
ユニラン ビクティニ ビクティニ ヒトカゲ フシギソウ
ユニラン ビクティニ ビクティニ ゼニガメ リザード
ユニラン ビクティニ ビクティニ ゼニガメ キャタピー
ユニラン ビクティニ ビクティニ ゼニガメ キャタピー

ヤドキング

ペラップ

アンノーン

ユニラン ビクティニ ビクティニ ヒトカゲ リザードン
ユニラン ビクティニ ビクティニ リザードン カメックス
ユニラン ビクティニ ビクティニ ゼニガメ キャタピー
ユニラン ビクティニ ビクティニ ゼニガメ キャタピー

ピカチュウ

ヤドラン

ポップォ アマルルガ ドンメル リザード カポエラー ドンメル ビクティニ

ペラップ

アンノーン

ユニラン ビクティニ ビクティニ ヒトカゲ フシギソウ
ユニラン ビクティニ ビクティニ ゼニガメ リザード
ユニラン ビクティニ ビクティニ ゼニガメ キャタピー
ユニラン ビクティニ ビクティニ ゼニガメ キャタピー

ヤドキング

ペラップ

ポップォ

ピカチュウ

ロトム

ポップォ

ポップォ プラスル ドンメル フシギダネ

ピカチュウ

アルセウスメタモンポッポドンメルフシギダネセレビィポッポサワムラーヤジロンドンメルフシギダネリザード
ヤドンヤドランポッポアマルルガドンメルフシギバナカポエラードンメルビクティニヤドンヤドランポッポ
アマルルガドンメルリザードカポエラードンメルビクティニペラップアンノーンユニランビクティニビクティニヒトカゲ
リザードンユニランビクティニビクティニリザードンカメックスユニランビクティニビクティニゼニガメ
キャタピーユニランビクティニビクティニゼニガメキャタピーユニランビクティニビクティニヒトカゲ
フシギソウユニランビクティニビクティニゼニガメリザードユニランビクティニビクティニゼニガメキャタピー
ユニランビクティニビクティニゼニガメキャタピーヤドキングペラップアンノーンユニランビクティニ
ビクティニヒトカゲリザードンユニランビクティニビクティニリザードンカメックスユニランビクティニ
ビクティニゼニガメキャタピーユニランビクティニビクティニゼニガメキャタピーピカチュウヤドラン
ポッポアマルルガドンメルリザードカポエラードンメルビクティニペラップアンノーンユニランビクティニ
ビクティニヒトカゲフシギソウユニランビクティニビクティニゼニガメリザードユニランビクティニ
ビクティニゼニガメキャタピーユニランビクティニビクティニゼニガメキャタピーヤドキングペラップ
ポッポピカチュウロトムポッポポッポプラスルドンメルフシギダネピカチュウ

```
'use strict';
let pidgey = 1;
while (pidgey <= 15) {
  if (pidgey % 3 === 0) {
    if (pidgey % 5 === 0) {
      console.log('FizzBuzz');
    } else {
      console.log('Fizz');
    }
  } else if (pidgey % 5 === 0) {
    console.log('Buzz');
  } else {
    console.log(pidgey);
  }
  pidgey = pidgey + 1;
}
```

```
1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz
```

まとめ・展望

1. 概要
2. 言語仕様
 - プログラム
 - リテラル
 - 演算
 - 出力
 - 変数
 - 条件分岐
 - 反復
3. FizzBuzz
4. まとめ・展望

まとめ・展望

- ポケモンの名前のみ記述可能なAltJSを作成
- 今後機会があれば実装したい
 - データ構造(配列など)
 - 型チェック
 - 識別子を全ポケモンに対応

参考文献

- ポケモン徹底攻略
https://yakkun.com/swsh/pokemon_en.htm
- ポケモンずかん
<https://zukan.pokemon.co.jp/>
- ポケモンだいすきクラブ
<https://www.pokemon.jp/>
- Pegjs GitHub
<https://github.com/pegjs/pegjs/blob/master/examples/javascript.pegjs>