

# Webプログラミング

★ 加藤大輔

# 目次

- 前回の課題のヒント
- DOMを使った表示操作
- onloadイベントを使った読み込み
- クリック後に反応するプログラム
- フォームの利用
- チェックボックスとラジオボタン

**前回の内容**  
**課題3の提出日は…5/19とします**

# 課題03(ヒント)

- 配布したプログラムを参考にして以下の実行画面を表示しなさい
- 今回最初のほうに用意した変数は
- `zaiko.kosuu = 0;`
- `zaiko.tanka = 125;`
- `zaiko.kingaku = 0;`

## (株)新座商事在庫管理

現在の在庫有高=0	最初在庫は無いので0
現在の在庫個数=100	100 個追加仕入したので 100
現在の在庫有高=12500	125 円×100 個の金額
現在の在庫個数=50	50 個売れたので数が減る
現在の在庫有高=6250	125 円×50 個の金額
現在の在庫個数=250	200 個追加仕入したので 250
現在の在庫有高=31250	125 円×250 個の金額
現在の在庫個数=220	30 個売れたので 220 に
現在の在庫有高=27500	125 円×30 個の金額



# 課題03(ヒント)

- <body>内で宣言した<script>
- `zaiko.aridaka();` → 現在の在庫有高 = 0      最初在庫は無いので 0
- `zaiko.tsuika(100);` → 現在の在庫個数 = 100      100 個追加仕入したので 100
- `zaiko.aridaka();` → 現在の在庫有高 = 12500      125 円 × 100 個の金額
- `zaiko.sakujo(50);` → 現在の在庫個数 = 50      50 個売れたので数が減る
- `zaiko.aridaka();` → 現在の在庫有高 = 6250      125 円 × 50 個の金額
- `zaiko.tsuika(200);` → 現在の在庫個数 = 250      200 個追加仕入したので 250
- `zaiko.aridaka();` → 現在の在庫有高 = 31250      125 円 × 250 個の金額
- `zaiko.sakujo(30);` → 現在の在庫個数 = 220      30 個売れたので 220 に
- `zaiko.aridaka();` → 現在の在庫有高 = 27500      125 円 × 30 個の金額

# 課題03(ヒント)

- 現在の在庫有高では...
- 在庫個数 × 在庫の単価
- で求めている

## (株)新座商事在庫管理

現在の在庫有高 = 0	最初在庫は無いので 0
現在の在庫個数 = 100	100 個追加仕入したので 100
現在の在庫有高 = 12500	125 円 × 100 個の金額
現在の在庫個数 = 50	50 個売れたので数が減る
現在の在庫有高 = 6250	125 円 × 50 個の金額
現在の在庫個数 = 250	200 個追加仕入したので 250
現在の在庫有高 = 31250	125 円 × 250 個の金額
現在の在庫個数 = 220	30 個売れたので 220 に
現在の在庫有高 = 27500	125 円 × 220 個の金額

# 課題03(ヒント)

- 従ってzeikin.aridaka = function()

の内部の計算式は以下の通りになる

```
zaiko.kingaku = zaiko.tanka * zaiko.kosuu;
```

あとはdocument.writeを使用して、出力例通りに

入力表示できれば良い

# DOMを使った表示操作



# DOMを使った表示操作

- ドキュメントオブジェクトモデル (DOM) は  
ウェブ文書のためのプログラミングインターフェイス  
ページを表現するため、プログラムが文書構造、スタイル、  
内容を変更することができます。  
DOM は文書をノードとオブジェクトで表現します。  
そうやって、プログラミング言語をページに接続することができます。

# DOMを使った表示操作

- 配布したhtmlの空欄に以下のプログラムを入力してください

```
<script>
var title = document.querySelector('#title');
var msg = document.querySelector('#msg');
title.textContent = "こんにちは";
msg.textContent = "表示した後に上書き出来る。";
</script>
```

- document.querySelector
- JavaScriptから任意のHTML要素を検出・取得することができる

# DOMを使った表示操作

メソッド  
変数 = document.querySelector ( 要素の指定 );  
オブジェクト

var `title` = document.querySelector( '#`title`' );  
`title`.textContent = "こんにちは";  
プロパティ

<h1 id="`title`">HELLO!</h1> 要素名の指定  
要素名からオブジェクトをつかむ  
その要素を変更

- document.querySelector
- JavaScriptから任意のHTML要素を検出・取得することができる

# DOMを使った表示操作

- <head>の方へ移動して確認してみよう

```
<!DOCTYPE html>
<html lang="ja">

<head>
<meta charset="UTF-8">
<title>HELLO</title>
<style>
body { font-size:14pt; font-weight:plain; }
h1 { color:white; padding:5px; font-size:24pt;background-color:red; }
</style>

<script>
function init{
var title = document.querySelector('#title');
var msg = document.querySelector('#msg');
title.textContent = "こんにちは";
title.textContent = "表示した後に上書きできる";
}
</script>

</head>

<body onload="init();">
<h1 id="title">HELLO!</h1>
<p id="msg">I am a boy.</p>
</body>

</html>
```



# DOMを使った表示操作

- Scriptは<head>の時点で使用することがあるが…
- 先に入力してしまうと変化が起きない

# onloadイベントを使った読み込み

- しかし、<body>内より<head>にまとめる方がいいので…
- 新たにonloadイベントを使用していきます！

HTMLページに含まれるすべてのリソース（スタイルシートや画像）の読み込みが完了した時点で発生するイベント

タイミングとしてはロード画面中

# onloadイベントを使った読み込み

- 次のプログラムに書き換えてください

```
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
<!DOCTYPE html>
<html lang="ja">

<head>
<meta charset="UTF-8">
<title>HELLO</title>
<style>
body { font-size:14pt; font-weight:plain; }
h1 { color:white; padding:5px; font-size:24pt;background-color:red;
</style>

<script>
function init(){
var title = document.querySelector('#title');
var msg = document.querySelector('#msg');
title.textContent = "こんにちは";
msg.textContent = "表示した後に上書きできる";
}
</script>

</head>

<body onload="init();">
<h1 id="title">HELLO!</h1>
<p id="msg">I am a boy.</p>
</body>

</html>
```

# クリック後に反応するプログラム

- <script>と<body>内の  
プログラムを  
書き換えてください

```
<!DOCTYPE html>
<html lang="ja">

<head>
<meta charset="UTF-8">
<title>HELLO</title>
<style>
body { font-size:14pt; font-weight:plain; }
h1 { color:white; padding:5px; font-size:24pt;background-color:red; }
</style>

<script>
function clickNow(){
var msg = document.querySelector('#msg');
msg.textContent = "クリックしましたね";
}
</script>
</head>

<body onload="init();">
<h1 id="title">HELLO!</h1>
<p id="msg">ボタンをクリックしてください。</p>
<button onClick="clickNow();">Click</button>
</body>

</html>
```



# フォームの利用

- 使用できる代表的な入力フォームの種類
- <text>、<checkbox>、<radio>、<file>、<submit>、<reset>など...



A screenshot of a web form with a light yellow background. The form contains several input fields and buttons. At the top, there is a text input field labeled '名前:'. Below it is a password input field labeled 'パスワード:'. The next line shows a series of radio buttons for '学年:' with options from '1年生' to '6年生'. Below that is a row of checkboxes for '好きな科目:' with options '国語', '英語', '算数', '理科', '社会', and '体育'. The next line is a file upload field labeled '宿題ファイル:' with a button 'ファイルを選択' and the text '選択されていません'. At the bottom are two buttons: '送信' (Submit) and 'リセット' (Reset).

名前:

パスワード:

学年: ☐ 1年生 ☐ 2年生 ☐ 3年生 ☐ 4年生 ☐ 5年生 ☐ 6年生

好きな科目: ☐ 国語 ☐ 英語 ☐ 算数 ☐ 理科 ☐ 社会 ☐ 体育

宿題ファイル:  選択されていません

# フォームの利用

- プログラムを  
書き換えてください

```
<!DOCTYPE html>
<html lang="ja">

<head>
<meta charset="UTF-8">
<title>HELLO</title>
<style>
body { font-size:14pt; font-weight:plain; }
h1 { color:white; padding:5px; font-size:24pt;background-color:red; }
</style>

<script>
function clickNow(){
var input1 = document.querySelector('#input1');
var value = input1.value;
var msg = document.querySelector('#msg');
msg.textContent = "こんにちは、"+value+"さん";
}
</script>
</head>

<body>
<h1 id="title">HELLO!</h1>
<p id="msg">あなたの名前は？</p>
<input type="text" id="input1">
<button onClick="clickNow();">Click</button>
</body>

</html>
```

# フォームの利用

- 成功例

**HELLO!**

あなたの名前は？



**HELLO!**

こんにちは、山田太郎さん

# プチまとめ

- 配布したプログラムを参考にして、以下の画像のように実行させなさい

HELLO!

あなたの名前は？

☒ 成人

☐ 女性

☒ 男性

Click

HELLO!

あなたは大人ですね。男性ですね。

☒ 成人

☐ 女性

☒ 男性

Click

HELLO!

まだ子供ですね男性ですね。

☐ 成人

☐ 女性

☒ 男性

Click