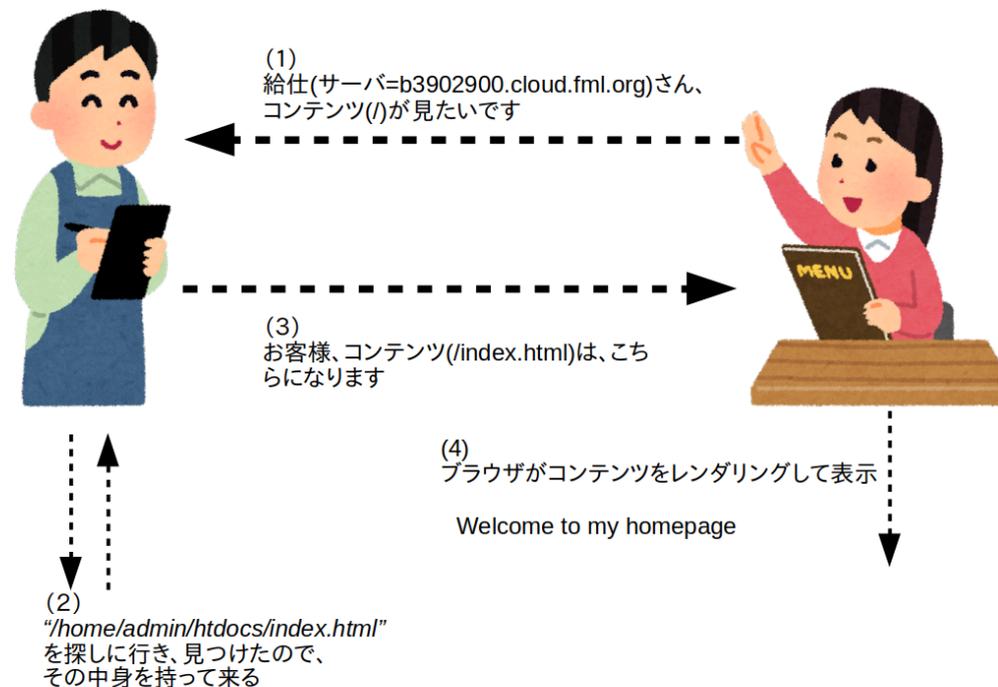


解説: HTML(1)

復習: ホームページのしくみ

- (1)
- (2)
- (3)
- (4)



前回のテキストより ... あえて空白にしてあります。分かるかな? って、図に説明が書いてありますけどね;-)

HTMLの基本(1): タグ

<P> 新しい段落(paragraph)を始めます。ブラウザは、改行など適切な見栄えになるようレンダリングを行います

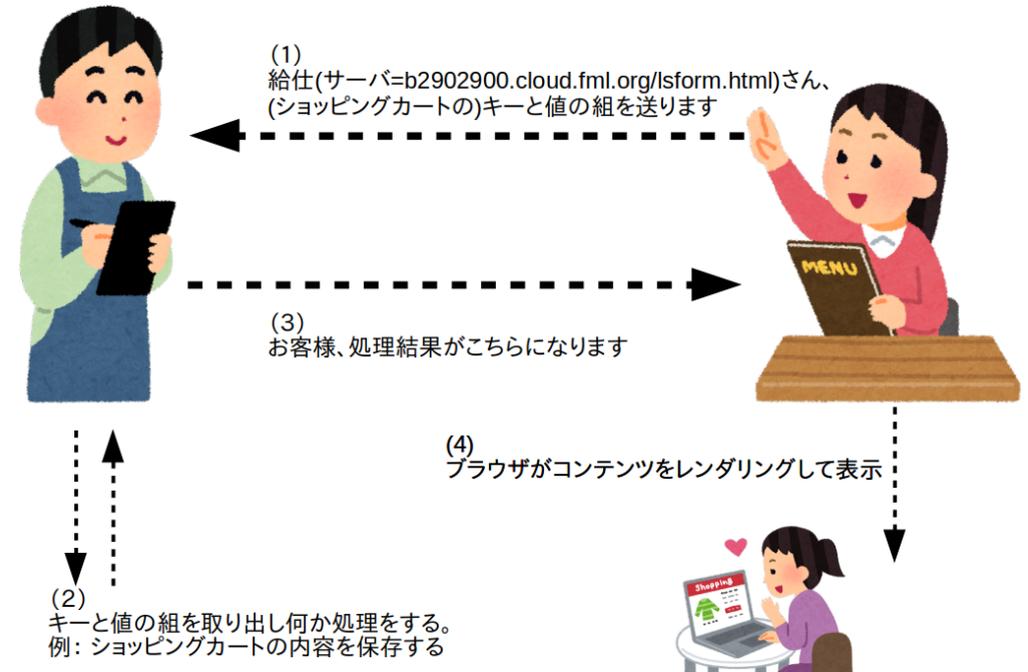
<P> このようにBタグではさむと、太い文字(ボールド体)で表示されます

- 単なるテキストです。ただしタグ(<tag>)と呼ばれる命令を含めることができます
- タグの始まり <tag> と終わり </tag> を書くと、それらのタグで挟まれた間だけで命令が有効になります
 - 閉じる方のタグは</(小なり+スラッシュ)>で始めます
- 動作確認だけ出来ればよい授業なので、**とりあえず<P>だけは覚えましょう。読みやすくなるから**

(脚注1) 真面目にHTML5をやると大変すぎるので初期のHTMLをやります (脚注2) この例で分かるように、HTMLのタグには文体の「構造」(P)と「装飾」(B)という異なる概念が一緒くたにされています。これが批判されて、厳密化され、現在のHTML5という読みづらい謎言語になりました。謎すぎるので授業ではやりません。ウェブデザイナーになりたい人は勉強してください

HTMLの基本(2a): FORM文(CGI)

- たとえばショッピングカートを作ること考えます。カートがあつかう情報とは「商品名」と「数量」の組です。組は複数可
- 元祖HTMLでは、こういうショッピングが出来ないため、新たに「キーと値の組をサーバに送る」約束事を作りました。この取り決めがCGI (CGI = Common Gateway Interface)
- WWWブラウザとサーバ両方がCGIをサポートしている必要があります (過去30年以上そうなので安心して使ってOK)



(脚注) ショッピングカートの最初のところだけです

復習: C言語の関数呼び出しを思い出そう

(ソースコードのイメージ)

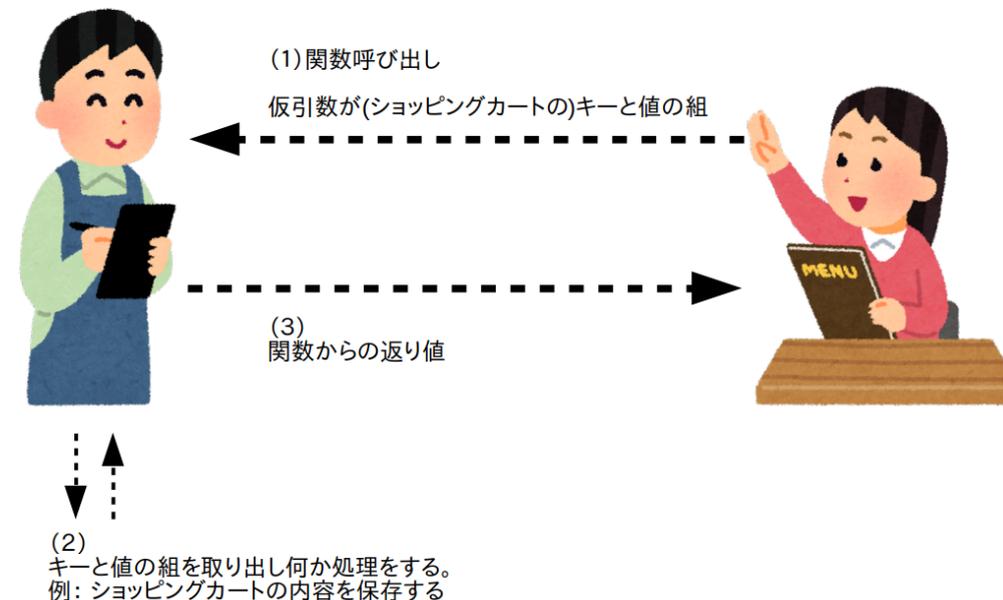
```
// カートに商品と数量を入れる
cart_add(item1, 1);
cart_add(item2, 2);
cart_add(item3, 3);

// 画面にショッピングカートの中身を表示する
cart_show();
```

- 処理を依頼するURLが関数に相当します
- だいぶ文法は異なりますが、C言語の関数呼び出しと同じですよ？
- CGIの場合、たいてい関数が別のPCで実行されるけど、**図**としては同じですよ？

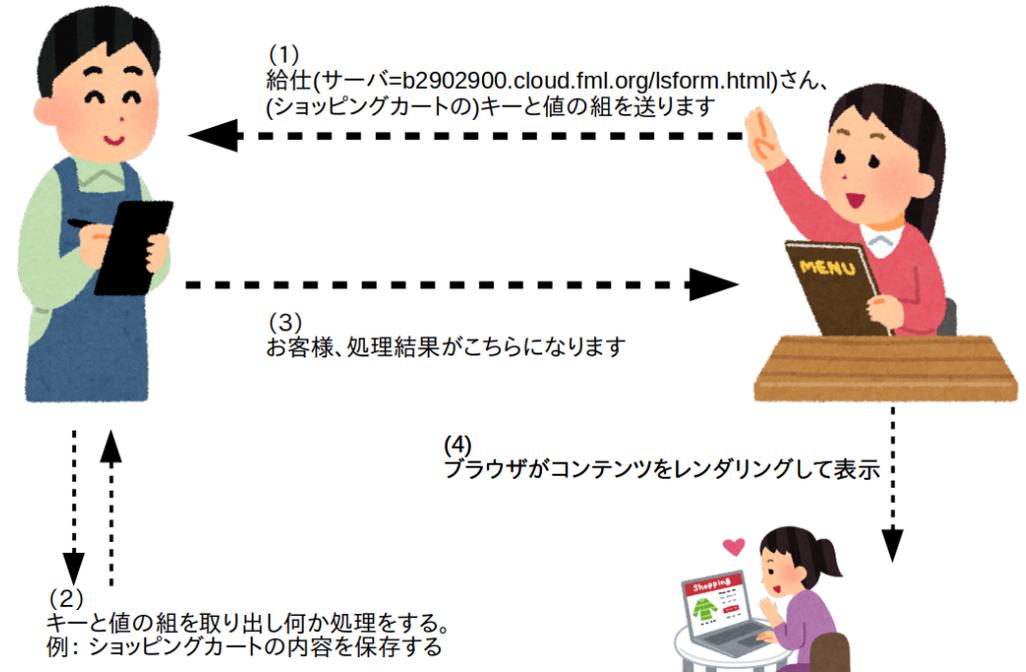
cart_add(キー,値) 関数

main 関数



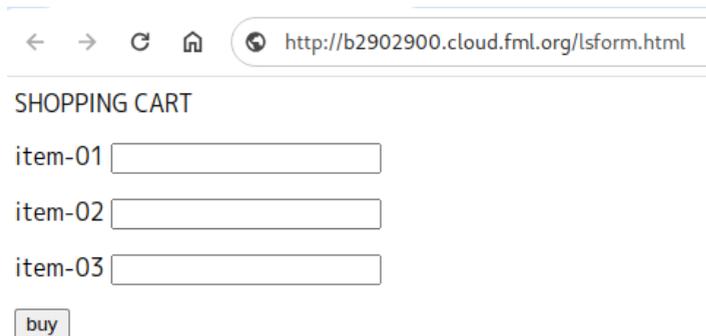
HTMLの基本(2b): FORM文(CGI)

- FORM文で、キーと値(「商品名」と「数量」)の組をサーバに送るHTMLを書けます
 - ブラウザはFORM文に沿って入力欄や送信ボタンなどを作成(レンダリング)します
 - ブラウザはキーと値の組をブラウザに送る
 - サーバ側では、キーと値の組を取り出す
- 上記までがCGIの仕様で、たいてい取り出した後、なんらかの処理をして結果を返すサーバの作りこみが、別途、必要です
(最近では、このサーバのことをWeb APIと呼ぶ)



(脚注) ショッピングカートの最初のところだけです

FORM文の体験例: <http://b2902900.cloud.fml.org/lform.html>



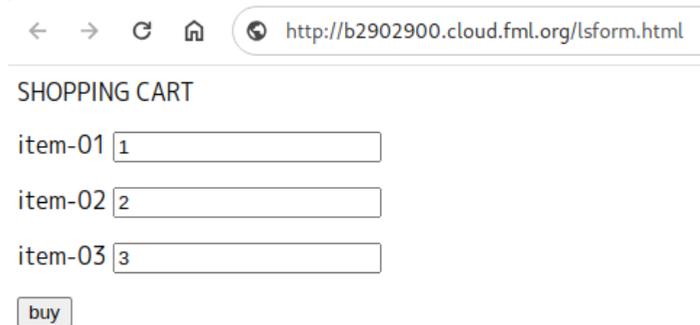
← → ↻ 🏠 <http://b2902900.cloud.fml.org/lform.html>

SHOPPING CART

item-01

item-02

item-03



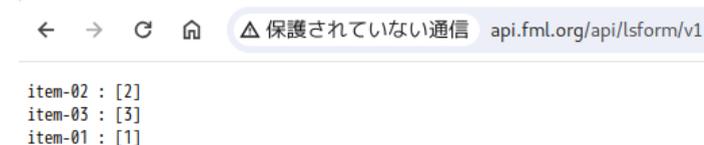
← → ↻ 🏠 <http://b2902900.cloud.fml.org/lform.html>

SHOPPING CART

item-01

item-02

item-03



← → ↻ 🏠 △ 保護されていない通信 api.fml.org/api/lform/v1

```
item-02 : [2]
item-03 : [3]
item-01 : [1]
```

(脚注1) このあと演習でlform.htmlをダウンロードして実際に体験してもらいます (脚注2) 図(右)の画面はサーバごとに異なります。これはlform.htmlのaction=URLで指定されているURL「<http://api.fml.org/>」で動いているサーバが、ブラウザから送られてきたキーと値の組を返している様子です。ちなみに、**演習のデモ等で使うサーバは、すべてGo言語で書かれています。これは「キーと値の組」を取り出して%v:%vでGo言語がよしなにフォーマットした表示になっています** (注: 内部では値をsliceで保持しているため[数字]と表示)

HTMLの基本(3a): lsform.htmlの解説 (画面の表示)



「画面に表示される文字」と「入力欄や送信ボタンの作り方」という2種類の話が混ざっていることに注意

HTMLの基本(3b): lsform.htmlの解説 (サーバの指定)

```
<form method="POST" action="http://api.fml.org/api/lsform/v1">
```

～省略～

```
</form>
```

- FORMタグ(<form>)で、データのやりとりの詳細設定を書きます
 - この部分はブラウザでレンダリングされませんが、
 - WWWサーバとのやりとりで重要な情報を書くところです
- タグのなかにはタグの属性等が書けます。スペース区切りで属性=値形式です
 - この例でも <FORM 属性1=値1 属性2=値2 ...> となっています
 - method=POST はHTTPの動作モードの指定と考えてください
FORM文では、たいていPOSTを指定するので、**授業ではmethod=POSTと覚えておいてOK**
 - action=URL が使うサーバのURLの指定です。ここは様々です。C言語で呼び出す関数を変えれば色々できるように、**URLを変えればイロイロな処理が可能となります**

HTMLの基本(3c): lsform.html (そのもの) 【オマケ】

```
<P>SHOPPING CART
<form method="POST" action="http://api.fml.org/api/lsform/v1">
  <P>item-01
    <input name="item-01" type="text">
  <P>item-02
    <input name="item-02" type="text">
  <P>item-03
    <input name="item-03" type="text">
  <P>
  <input type="submit" value="buy">
</form>
```

- これが、このあとダウンロードするlsform.htmlです

HTMLの基本(3d): lsform.htmlの読み方 【オマケ】

```
<P>SHOPPING CART                                <!-- ここは普通に表示されます(改行あり) -->
<form method="POST" action="Web APIサーバのURL"> <!-- methodはHTTPの動作モードのようなもの,大抵POST -->
  <P>キーの名称1                                <!-- ここは普通に表示されます(改行あり) -->
    <input name="キーの名称その1" type="text">    <!-- 入力欄としてレンダリングされます -->
  <P>キーの名称2                                <!-- ここは普通に表示されます(改行あり) -->
    <input name="キーの名称その2" type="text">    <!-- 入力欄としてレンダリングされます -->
  <P>キーの名称3                                <!-- ここは普通に表示されます(改行あり) -->
    <input name="キーの名称その3" type="text">    <!-- 入力欄としてレンダリングされます -->
  <P>                                             <!-- 改行するために無駄にPを入れました -->
  <input type="submit" value="送信ボタンの表示名"> <!-- 送信ボタンとしてレンダリングされます -->
</form>
```

- <!-- コメント --> で説明しています。 <P>~ の部分がないと箱が並ぶだけでページの意味が分かりません。「説明を書く元祖HTML」と「FORM文が作る入力欄」の両方を混ぜこぜで書いていきます
- 「キーの名称1」を「変数名1」と考えると、関数呼び出しのイメージと重なります