

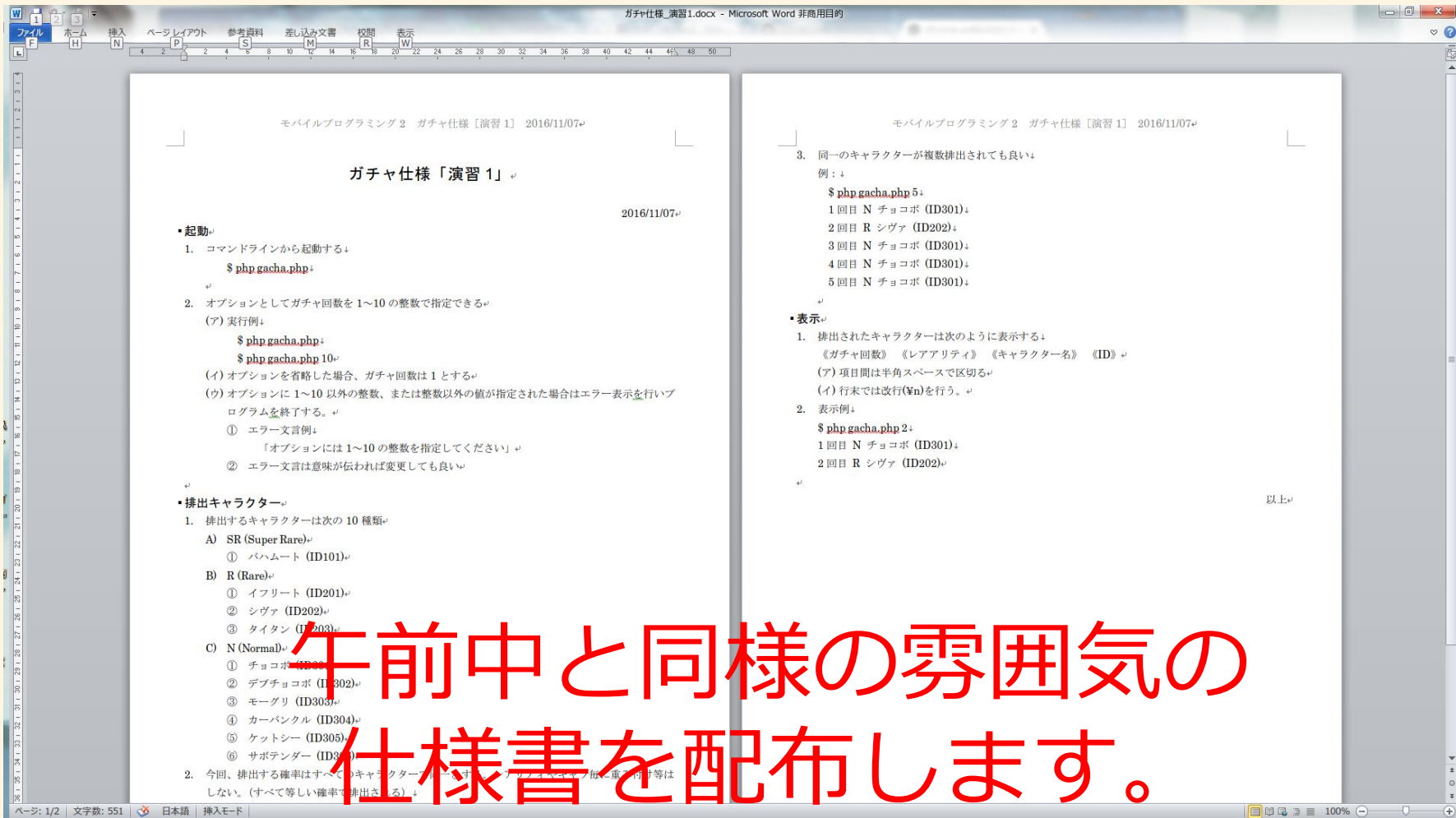
モバイル プログラミング 実習2

午後

- ネットワーク基礎
 - HTTPヘッダ
 - Cookie
- HTTP演習
 - ファイル処理

【予告】 来週の授業

仕様書を配布 チャットを作成



GitHubのアカウントを 各自で準備

 [Personal](#) [Open source](#) [Business](#) [Explore](#)

[Pricing](#) [Blog](#) [Support](#)

[Sign in](#) [Sign up](#)

How people build software

Millions of developers use GitHub to build personal projects, support their businesses, and work together on open source technologies.

Use at least one letter, one numeral, and seven characters.[Sign up for GitHub](#)By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy policy](#). We'll occasionally send you account related emails.

Welcome home, developers

GitHubからコードを提出

This repository Search Pull requests Issues Gist

katsube / neec Unwatch 3

<> Code Issues 52 Pull requests 0 Projects 0 Wiki Pulse Graphs Settings

G014A1348 TM (Chat) #2

Open katsube opened this issue 3 days ago · 0 comments

katsube commented 3 days ago neec owner

授業中に作成したソースコードを提出してください。
必ず授業中には提示した「合言葉」を記載すること。

※自分のIssueかよくタイトルを確認してください。
※このIssueへ提出することで出席となります。

katsube added this to the Chat milestone 3 days ago

Write Preview AA B i “ < > ☰ ☷ ☹ ↶ @ 📌

Leave a comment

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Styling with Markdown is supported

Close issue Comment

Signed in as katsube

- Your profile
- Your stars
- Explore
- Integrations
- Help
- Settings
- Sign out

Project: None yet

Labels: None yet

Milestone: Chat

Assignees: No one—assign yourself

1 participant

Notifications: Unsubscribe

You're receiving notifications because you authored the thread.

来週は「提出＝出席」とします

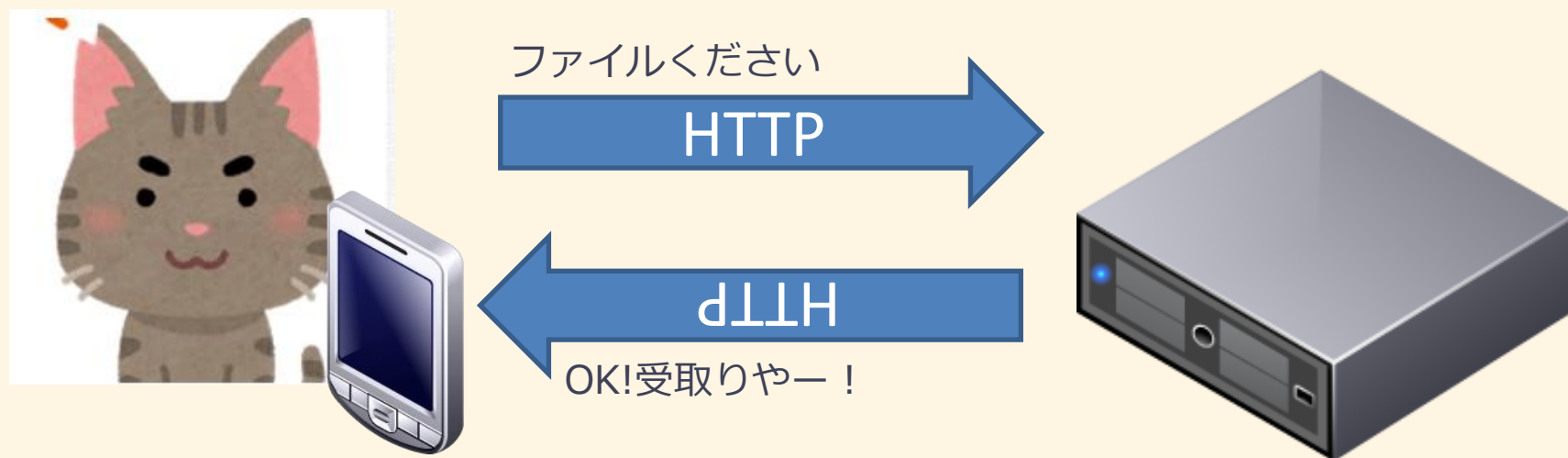
前回の復習

ネットワーク基礎

その4

HTTP メソッド

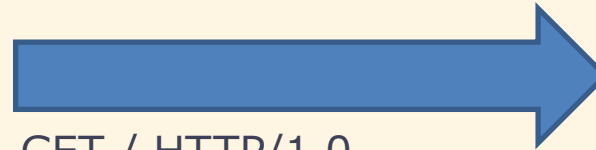
プロトコルとは何か？



HTTP



接続



GET / HTTP/1.0



HTTP/1.1 200 OK

Server: nginx

Date: Sun, 09 Oct 2016
13:19:36 GMT

Content-Type: text/html;
charset=UTF-8

Connection: close

telnet で http を手打ちする

クライアントから入力

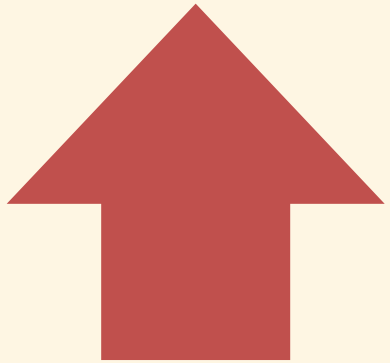
```
$ telnet www.yahoo.co.jp 80  
GET / HTTP/1.0
```

サーバからの返却

```
HTTP/1.1 200 OK  
Server: nginx  
Date: Sun, 09 Oct 2016 13:19:36 GMT  
Content-Type: text/html; charset=UTF-8  
Connection: close  
(以降HTML)
```

メソッド

```
$ telnet www.yahoo.co.jp 80  
GET / HTTP/1.0
```



この部分が「メソッド」

メソッド

- メソッドは「**コマンド**」のような物
- 例えば
 - **GET / POST**
 - ファイルをください
 - **HEAD**
 - ファイルの情報をください。
(中身はいらないです)

メソッド

- その他にも

- **PUT**

- こちら(クライアント)からファイルを送信するのでサーバに保存してください。

- **DELETE**

- サーバ上のファイルを削除してください。

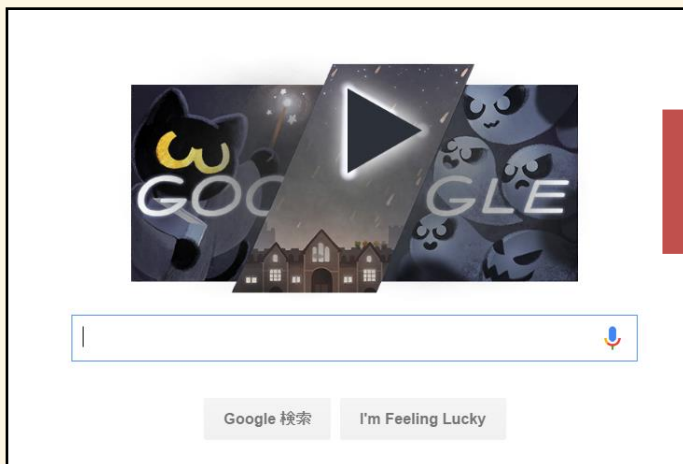
※他にもありますが、ひとまずこの5つを覚えておけば良いです。

GETとPOSTの違い

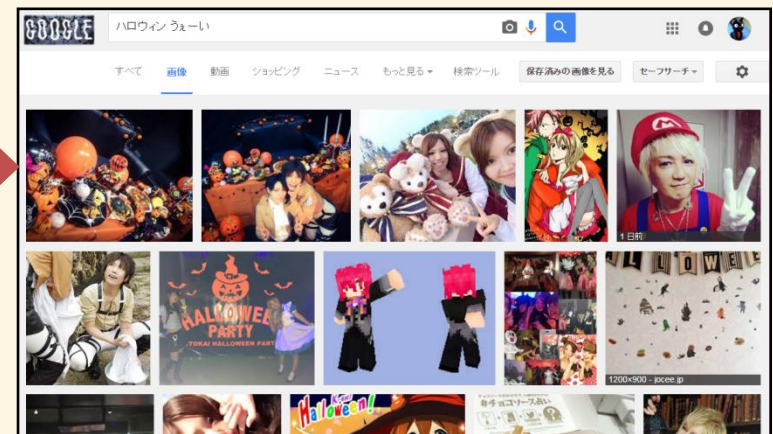
GET/POSTの使い分け

- GETとPOSTを使い分ける必要が出てくるのは、**サーバに値を渡すとき**です。

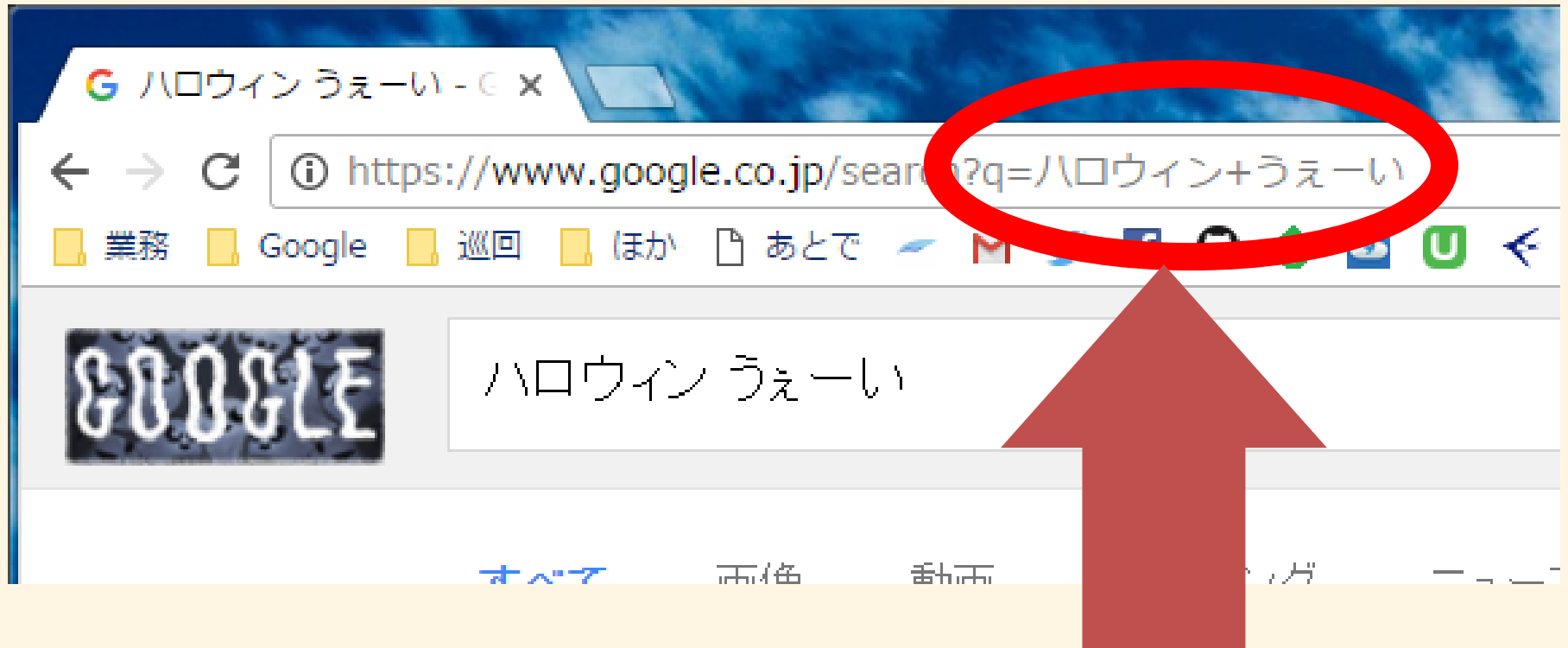
入力



入力した値によって
結果を変えたい



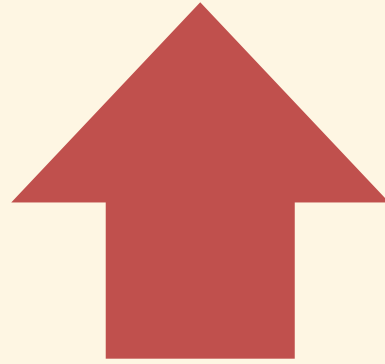
URLにクエリーを追加



この部分が値。
「クエリー」と呼びます。

手打ちする際も追加OK

```
$ telnet www.google.co.jp 80  
GET /search?q=cat HTTP/1.0
```



telnetを利用した場合も
同様に記述できます。

GETの問題点

- **文字列長の制限**

- HTTPの仕様上は制限ない。ブラウザの実装による問題。
- 概ね2000byte以内。

- **セキュリティの問題**

- URLはWebサーバのアクセスログなどに保存されます。
- 例えば**個人情報**や**パスワード**などがクエリーにあると一緒に記録されてしまう。

POSTを手打ちする

```
$ telnet www.example.com 80  
POST /receive.php HTTP/1.0  
Content-Length: 1234  
  
q=cat&foo=bar&hoge=huga...
```

※クエリーの書式はこの後説明を行います。

GET

- **利点**

- ブラウザのURL入力欄に指定できる
 - A要素(リンクタグ)での指定も可能
- ブラウザのキャッシュが効く

- **制約**

- 文字列長の制限
- セキュリティ

POST

• 利点

- 大きなサイズのデータが送信できる
- セキュリティ上安心

※いずれもGETと比較した場合。

• 制約

- ブラウザの入力欄で指定できない
- キャッシュが効かない
 - 基本的に毎回送信する必要がある

クエリーの書式

クエリーの書式

クエリーが一つ

name=value

クエリーが複数

q1=value1&q2=value2

q1=value1&q2=value2&q3=value3

&で結合することでいくらでも指定できます

URLエンコード

- 以下の値の利用には**エンコード**が必要
 - URLで使用できない文字
 - マルチバイト文字
 - バイナリ

※ここでは超ざっくり説明しています。

URLエンコード

and those reserved characters not acting as delimiters, define each component's identifying data.

2.1. Percent-Encoding

A percent-encoding mechanism is used to represent a data octet in a component when that octet's corresponding character is outside the allowed set or is being used as a delimiter of, or within, the component. A percent-encoded octet is encoded as a character triplet, consisting of the percent character "%" followed by the two hexadecimal digits representing that octet's numeric value. For example, "%20" is the percent-encoding for the binary octet "00100000" (ABNF: %x20), which in US-ASCII corresponds to the space character (SP). [Section 2.4](#) describes when percent-encoding and decoding is applied.

pct-encoded = "%" HEXDIG HEXDIG

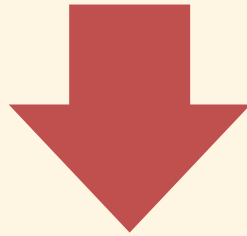
The uppercase hexadecimal digits 'A' through 'F' are equivalent to the lowercase digits 'a' through 'f', respectively. If two URIs differ only in the case of hexadecimal digits used in percent-encoded octets, they are equivalent. For consistency, URI producers and normalizers should use uppercase hexadecimal digits for all percent-encodings.

2.2. Reserved Characters

URIs include components and subcomponents that are delimited by characters in the "reserved" set. These characters are called "reserved" because they may (or may not) be defined as delimiters by the generic syntax, by each scheme-specific syntax, or by the

URLエンコード

肉



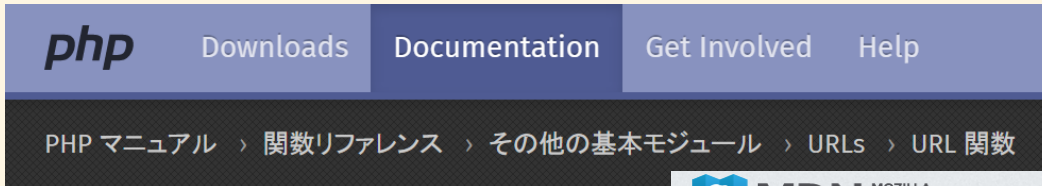
%E8%82%89

URLエンコード

name=%E8%82%89

※name部分にマルチバイト文字を使った場合も、同様にエンコードが必要です。
※なるべく左辺には使わないようにするのがおすすめです。

URLエンコード



urlencode

(PHP 4, PHP 5, PHP 7)

urlencode — 文字列を URL エンコードする

説明

```
string urlencode ( string $str )
```

この関数は、URL の問い合わせ部分に使用する文

The image is a screenshot of the MDN (Mozilla Developer Network) documentation page for the JavaScript function `encodeURIComponent()`. The page is in Japanese. At the top, there's a navigation bar with 'MDN' logo, 'MOZILLA DEVELOPER NETWORK', and a 'ログイン' (Login) button. Below the navigation bar, there's a search bar and a list of categories: 'ウェブ技術', 'MOZILLA DOCS', '開発者ツール', and 'フィードバック'. The main content area shows the function name `encodeURIComponent()` in large text. Below it, there's a section titled 'この記事内' (In this article) with a plus sign. The description states that the function encodes a string into a URI (Uniform Resource Identifier) using UTF-8 encoding, replacing characters with their escaped sequences. The '構文' (Syntax) section shows the function signature: `encodeURIComponent(str);`. The '引数' (Arguments) section lists the parameter `str` as a string to be encoded. The '戻り値' (Return value) section states that it returns a new string with the characters escaped.

<http://php.net/manual/ja/function.urlencode.php>

https://developer.mozilla.org/ja/docs/Web/JavaScript/Reference/Global_Objects/encodeURIComponent

URLエンコード

- URL中に使用できない文字やバイナリはあらかじめエンコードする必要がある。
- URLエンコード/デコードを行う手段が提供されている言語では、それらを用いる。

動的な Webページ



演習1. MAP表示

配列

```
$map= [  
    [0, 0, 0, 0]  
    , [0, 1, 0, 0]  
    , [0, 0, 0, 0]  
    , [2, 2, 2, 2]  
];
```



0

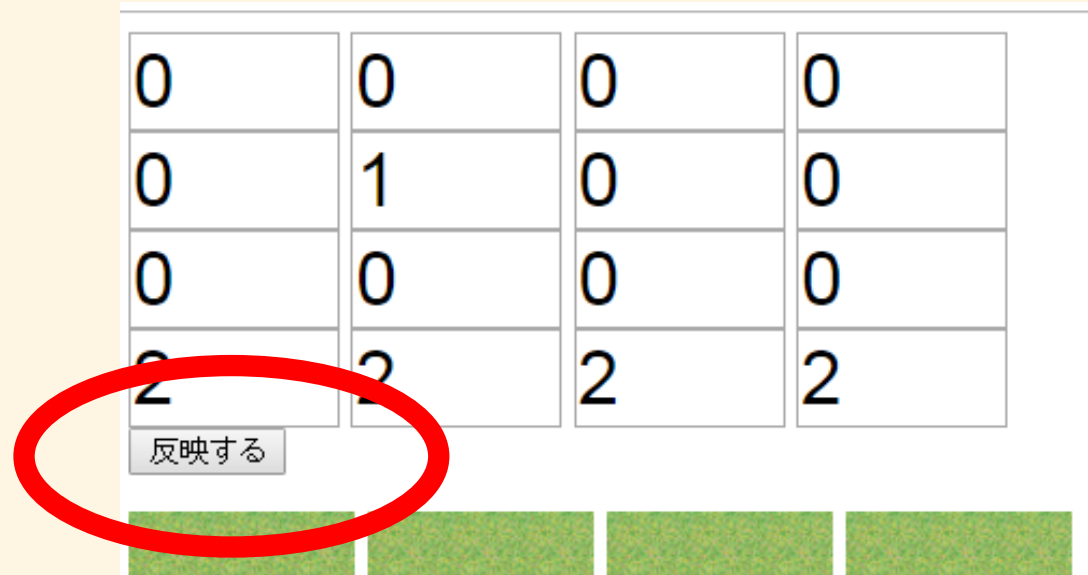
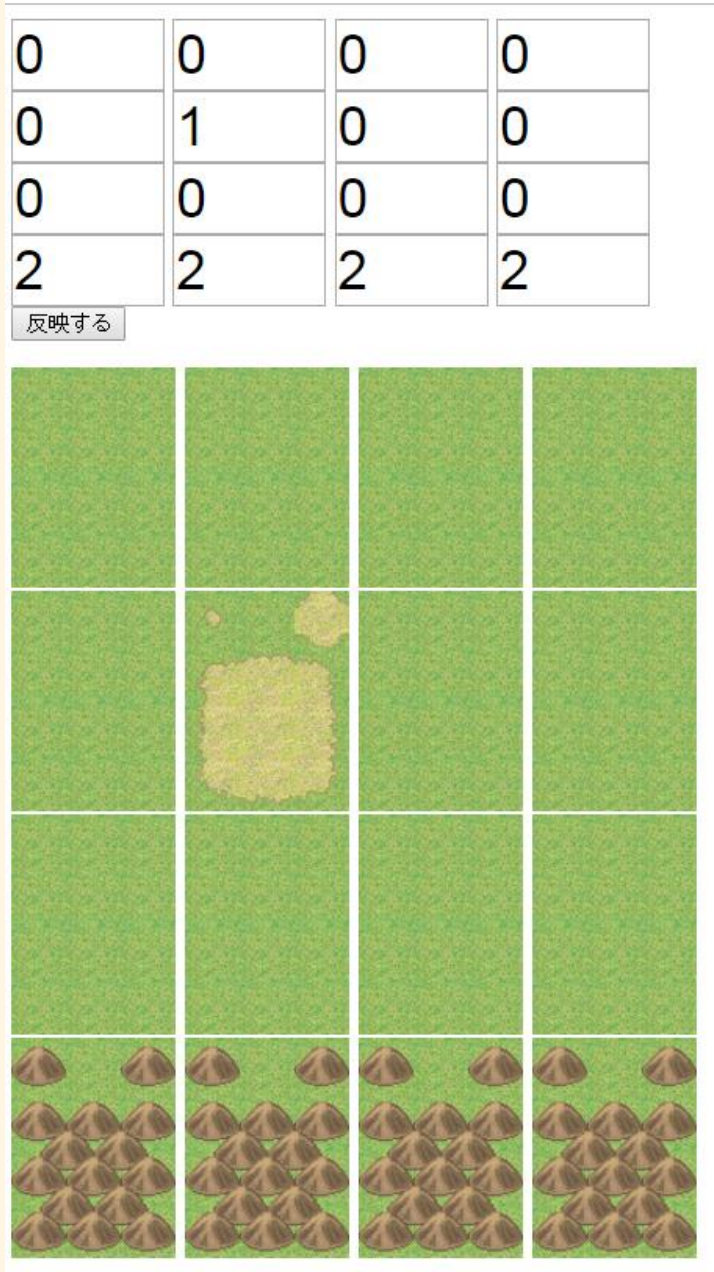


1



2

演習2. MAPエディタ



ボタンを押すと
MAPに反映

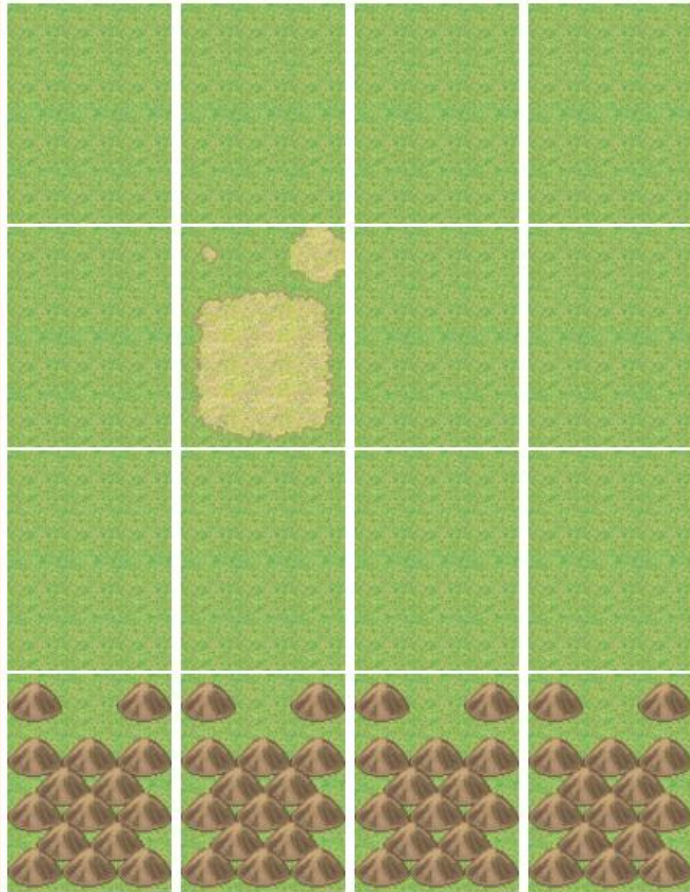
PHPにおける ファイル処理



演習1. MAPエディタ（保存）

0	0	0	0
0	1	0	0
0	0	0	0
2	2	2	2

反映する



反映ボタンを押すと
MAPを表示しつつ、
サーバ上のファイル
に
保存するようにしま
しょう。

シリアライズ serialize

php

Downloads

Documentation

Get Involved

Help

Search

PHP マニュアル > 関数リファレンス > 変数・データ型関連 > 変数操作 > 変数操作 関数

« print

serialize

Change language: Japanese ▼

[Edit](#) [Report a Bug](#)

(PHP 4, PHP 5, PHP 7)

serialize — 値の保存可能な表現を生成する

説明

```
string serialize ( mixed $value )
```

値の保存可能な表現を生成します。

型や構造を失わずに PHP の値を保存または渡す際に有用です。

シリアル化された文字列を PHP の値に戻すには、[unserialize\(\)](#) を使用してください。

パラメータ

変数操作 関数

[boolval](#)

[debug_zval_dump](#)

[doubleval](#)

[empty](#)

[floatval](#)

[get_defined_vars](#)

[get_resource_type](#)

[gettype](#)

[import_request_variables](#)

[intval](#)

[is_array](#)

[is_bool](#)

[is_callable](#)

[is_double](#)

[is_float](#)

[is_int](#)

[is_integer](#)

[is_long](#)

[is_null](#)

[is_numeric](#)

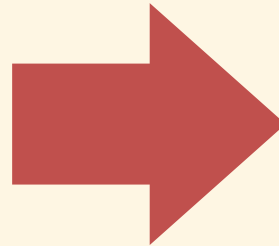
[is_object](#)

演習2. MAPエディタ（ログイン認証）

ログイン

ID:

PW:



0	0	0	0
0	1	0	0
0	0	0	0
2	2	2	2

The map is a 4x4 grid. The top three rows are green. The bottom row is filled with brown mounds. A yellow patch is visible in the second row, second column.

演習3. MAPエディタ（IDを記録）



A mockup of a login form. It has a title 'ログイン' (Login) in bold black text. Below the title are two input fields: the first is labeled 'ID:' and the second is labeled 'PW:'. Below the 'PW:' field is a button labeled 'ログイン' (Login).

ログインIDをブラウザに記録させましょう。

ここではCookieを利用して実装してみましよう。

演習4. MAPエディタ（セッション）



A login form titled "ログイン" (Login). It contains two input fields: "ID:" and "PW:". Below the "PW:" field is a button labeled "ログイン" (Login).

ログイン

ID:

PW:

セッションに対応しましょう。

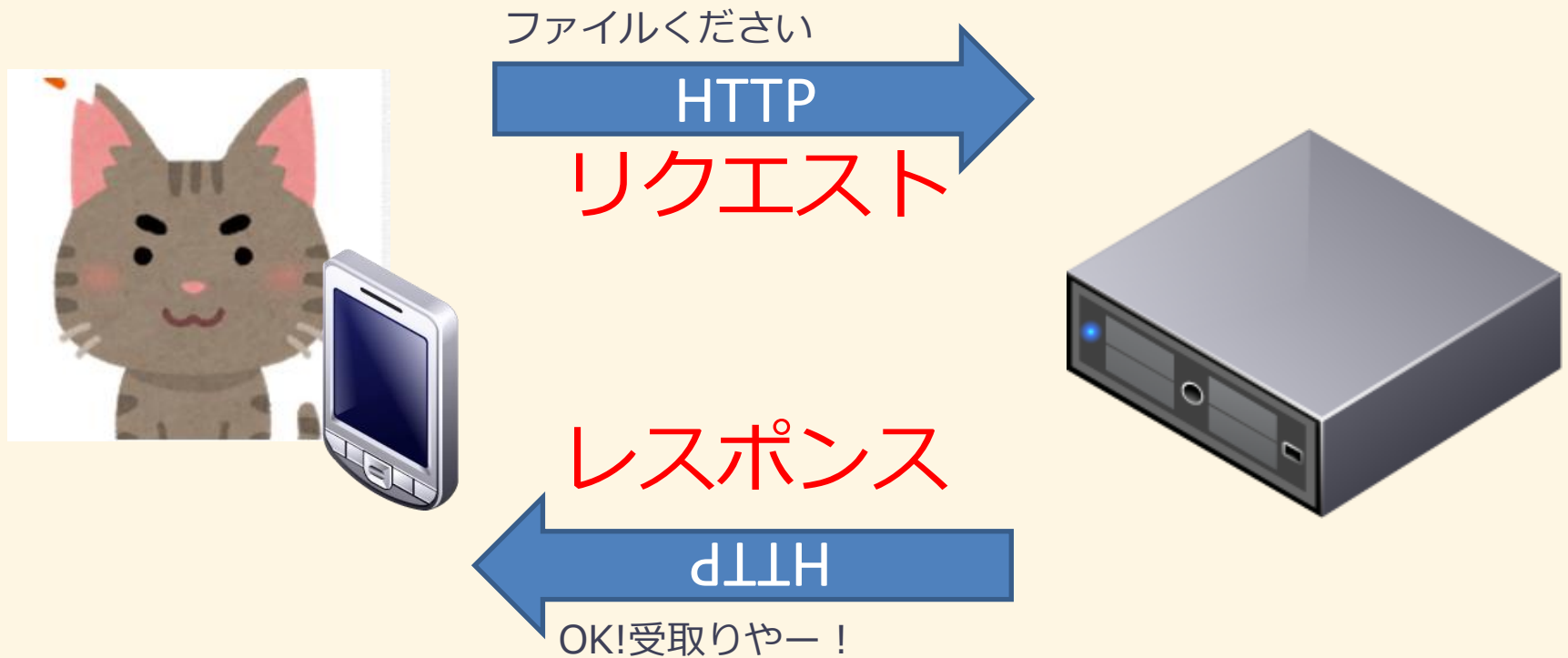
ログイン済みの状態でこの画面に来たら、自動的にMAPをエディタへ遷移させましょう。

ネットワーク基礎

その5

HTTPヘッダ

リクエスト・レスポンス



リクエスト(GET)

```
$ telnet 127.0.0.1 80  
GET /hello.html HTTP/1.0
```

リクエスト(POST)

```
$ telnet 127.0.0.1 80
```

```
POST /hello.php HTTP/1.0
```

```
foo=bar&hoge=huga&hoge=huga&h  
oge=huga&hoge=huga&hoge=huga
```

リクエスト

```
$ telnet 127.0.0.1 80  
GET /hello.html HTTP/1.0
```

} リクエスト
ヘッダ

```
$ telnet 127.0.0.1 80  
POST /hello.php HTTP/1.0
```

(空行)

```
foo=bar&hoge=huga&hoge  
=huga&hoge=huga&hoge=  
huga&hoge=huga
```

} リクエスト
ヘッダ

} リクエスト
ボディ

レスポンス

```
[neec@localhost ~]$ telnet 127.0.0.1 80
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
GET /hello.html HTTP/1.0

HTTP/1.1 200 OK
Date: Sun, 06 Nov 2016 11:47:26 GMT
Server: Apache/2.4.6 (CentOS) PHP/5.6.26
Last-Modified: Sun, 06 Nov 2016 11:47:13 GMT
ETag: "59-540a077da6f5c"
Accept-Ranges: bytes
Content-Length: 89
Connection: close
Content-Type: text/html; charset=UTF-8
(空行)
<html>
<head>
    <title>Hello</title>
</head>
<body>
    <h1>HelloWorld!</h1>
</body>
</html>
Connection closed by foreign host.
```

レスポンス
ヘッダ

レスポンス
ボディ

リクエスト・レスポンス

リクエスト

HTTP



ヘッダ

ボディ



ヘッダ

ボディ

HTTP

レスポンス



GoogleChromeでも確認できる

Network

The screenshot shows a Google Chrome browser window displaying the Yahoo! JAPAN homepage. The address bar shows the URL `www.yahoo.co.jp`. The developer tools are open on the right side, with the **Network** tab selected. A red circle highlights the **Network** tab, and another red circle highlights the **www.yahoo.co.jp** entry in the list of requests. The details for this request are visible on the right, showing a **GET** request to `http://www.yahoo.co.jp/` with a **Status Code** of **200 OK**. The response headers include **Cache-Control**, **Connection**, **Content-Encoding**, **Content-Type**, **Date**, **Expires**, **P3P**, **Pragma**, **Server**, **Transfer-Encoding**, **Vary**, **X-Content-Type-Options**, **X-Frame-Options**, **X-XRDS-Location**, and **X-XSS-Protection**. The request headers include **Accept**, **Accept-Encoding**, **Accept-Language**, **Connection**, and **Cookie**.

フ

ファイル名を

クリック

ヘッダの 種類

リクエスト

▼ Request Headers [view parsed](#)

GET / HTTP/1.1

Host: www.nicovideo.jp

Connection: keep-alive

Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/54.0.2840.71 Safari/537.36

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

DNT: 1

Accept-Encoding: gzip, deflate, sdch

Accept-Language: ja,en-US;q=0.8,en;q=0.6

Cookie: nicosid=1478436297.44177100

レスポンス

▼ Response Headers

view parsed

HTTP/1.1 200 OK

Date: Sun, 06 Nov 2016 12:46:14 GMT

Server: Apache

Vary: Host,Accept-Encoding

x-niconico-authflag: 0

Content-Encoding: gzip

Access-Control-Allow-Credentials: true

Content-Length: 20639

Connection: close

Content-Type: text/html; charset=UTF-8

Content-Language: ja

Cookie

Cookieとは？

リクエスト

HTTP



次アクセスするときに、
この情報をそのまま返
して！

name=tarou



HTTP

レスポンス



Cookieとは？

リクエスト

HTTP

name=tarou



HTTP

レスポンス

サーバからクライアントへ渡すには レスポンスヘッダに含める

```
$ telnet www.twitter.com 80
```

```
GET / HTTP/1.0
```

```
host: twitter.com
```

```
HTTP/1.0 301 Moved Permanently
```

```
content-length: 0
```

```
date: Sun, 06 Nov 2016 12:34:59 GMT
```

```
location: https://twitter.com/
```

```
server: tsa_a
```

```
set-cookie: guest_id=v1%3A147843569907140709;
```

```
Domain=.twitter.com; Path=/; Expires=Tue, 06-Nov-2018 12:34:59  
UTC
```

```
x-connection-hash: 0c2a5bd193d3b1a471843852512176aa
```

```
x-response-time: 3
```


クライアントからサーバへ渡すときも リクエストヘッダに含める

```
$ telnet www.twitter.com 80
```

```
GET / HTTP/1.0
```

```
host: twitter.com
```

```
Cookie: guest_id=v1%3A147843569907140709
```

```
HTTP/1.0 301 Moved Permanently
```

```
content-length: 0
```

```
date: Sun, 06 Nov 2016 12:34:59 GMT
```

```
location: https://twitter.com/
```

```
server: tsa_a
```

```
set-cookie: guest_id=v1%3A147843569907140709;
```

```
Domain=.twitter.com; Path=/; Expires=Tue, 06-Nov-2018 12:34:59  
UTC
```

```
x-connection-hash: 0c2a5bd193d3b1a471843852512176aa
```

```
x-response-time: 3
```

PHPでCookie



Cookieをセット

php

Downloads

Documentation

Get Involved

Help

Search

PHP マニュアル › 関数リファレンス › その他のサービス › ネットワーク
› ネットワーク 関数

« pfsockopen

setraw

setcookie

Change language: Japanese

[Edit](#)

[Report a Bug](#)

(PHP 4, PHP 5, PHP 7)

setcookie — クッキーを送信する

説明

```
bool setcookie ( string $name [, string $value = "" [, int $expire =  
0 [, string $path = "" [, string $domain = "" [, bool $secure =  
false [, bool $httponly = false ]]]]] )
```

setcookie() は、その他のヘッダ情報と共に 送信するクッキーを定義します。 ほ

ネットワーク 関数

[checkdnsrr](#)

[closelog](#)

[define_syslog_variables](#)

[dns_check_record](#)

[dns_get_mx](#)

[dns_get_record](#)

[fsockopen](#)

[gethostbyaddr](#)

[gethostbyname](#)

[gethostbyname_l](#)

[gethostname](#)

[getmxrr](#)

[getprotobyname](#)

[getprotobyname](#)

Cookieを取得

php

Downloads

Documentation

Get Involved

Help

Search

PHP マニュアル › 言語リファレンス › 定義済の変数

« \$_ENV

\$php_err

\$_COOKIE

Change language: Japanese

Edit

Report a Bug

\$HTTP_COOKIE_VARS [非推奨]

(PHP 4 >= 4.1.0, PHP 5, PHP 7)

\$_COOKIE -- \$HTTP_COOKIE_VARS [非推奨] — HTTP クッキー

説明

現在のスクリプトに HTTP クッキーから渡された変数の連想配列です。

`$HTTP_COOKIE_VARS` は同じ情報を持っていますが、これは [スーパーグローバル](#) ではありません (`$HTTP_COOKIE_VARS` と `$_COOKIE` は違う変数であり、PHP はそれぞれ別に扱います)。

変更履歴

定義済の変数

スーパーグローバル

`$GLOBALS`

`$_SERVER`

`$_GET`

`$_POST`

`$_FILES`

`$_REQUEST`

`$_SESSION`

`$_ENV`

» `$_COOKIE`

`$php_errormsg`

`$HTTP_RAW_POST_DATA`

`$http_response_header`

`$argc`

`$argv`