ソフトウェア開発２
―はじめに＆周辺技術紹介―

鎌田十三郎

評価について

- 試験およびプログラムレポートにより評価する
- 出席： 評価に直結させませんが、参考のためにとっています
- 課題B: 標準課題もしくは自由課題を一つ提出
  - Web/Network もしくは Thread を扱った課題であること
  - ソフトウェア開発1のみ単位を取る場合は、GUI 課題のみでOK
- 試験
  - 12月中にソフトウェア開発1・2をまとめて実施予定
  - 1月は課題に時間を当ててもらう予定

講義の趣旨

演習を通してソフトウェア開発を経験しながら、
実用的なアプリケーションを構築するための
先進的なプログラミング技術を学習する

- ソフトウェア開発2の目標
  Web アプリケーション開発を題材に
  - ネットワーク接続に関する基本トピック
  - 非同期処理に関するクライアント技術
  - マルチスレッドを用いたサーバ技術

  を学習する
- プログラミング課題では、いくつかのトピックを取り組んでいればOK

注意事項＆アナウンス

- 分からないことを放置せず、早めに質問・対処する
  - 定期的に質問時間などを設ける予定
  - 遅れている人は、授業時間外にもプログラミングを頑張ってください
- カンニング行為（プログラムコピーとか）厳禁
- 利用プログラミング言語は、基本 Java 言語
  - サポートはしないですが、他のプログラミング課題を用いて課題に取り組んでもらっても構いません。

連絡先

- 簡単な質問は Slack で応対
- 皆さんの状況を確認しながら対応しないといけない案件は、Zoom で応対
- 成績に関する問い合わせなどは、BEEF もしくは Zoom（学生証などを見せてもらうかも）経由で応対

事前事後学習について

- 高度な概念を、プログラミング文面みただけで理解するのは、結構大変です。
  色々書いて、動かしてみて、理解が深まるものだと思います。しっかり手を動かすこと。
- 技能が身についていない場合は、必要に応じて授業時間外にも課題などに取り組んでください。

スケジュール

- ～12月: 講義中心＋試験
  - ネットワーク・Web をめぐる簡単な技術紹介
  - 先進的 I/O
  - ネットワーク接続、Webアクセス、非同期問い合わせ
  - マルチスレッドの基本
  - クラウドデータベース
- プログラミング課題
  - 12月中に、課題で構築するアプリケーション、実装イメージを煮詰めましょう
  - 実装は1月でOK

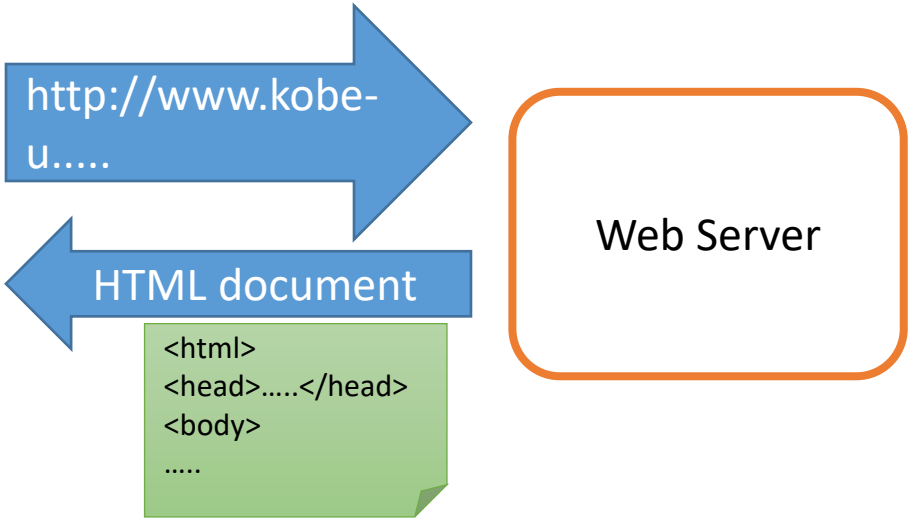
今日のお話

- 皆さんが普段使っている Web アプリなどのベースとなる技術の紹介
  （今後教える各種技術との関係を把握するため）

- その後は、Web Text でプログラムを扱った講義

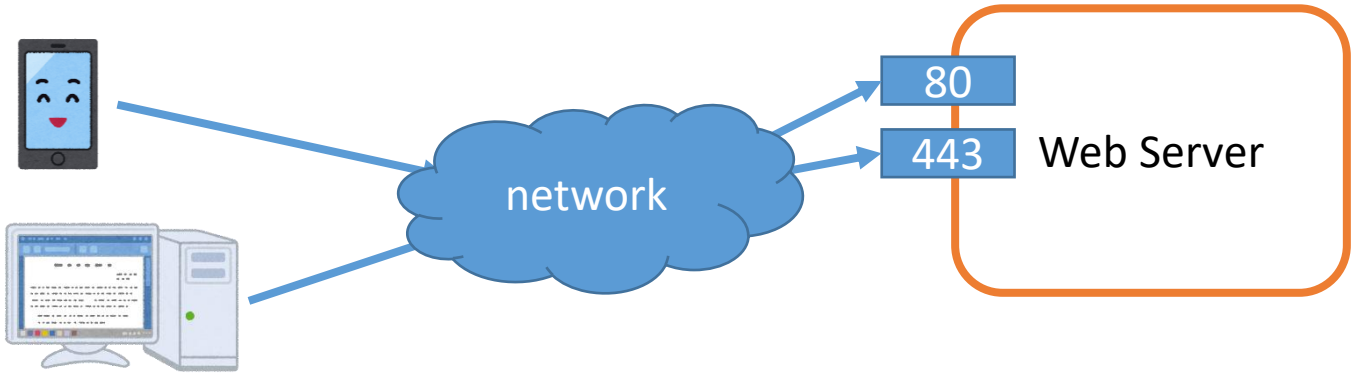
手始めに、
Web サーバ

- HTTP request を処理
  - URLの例: http://www.kobe-u.ac.jp/info/public-relations/uribo/index.html
  - 指定サーバの指定ポート(標準: 80)で待ち受け
  - URL に応じた文書を返す（普通はHTML 文書）
- サーバ: URL に応じたテキストを返す関数



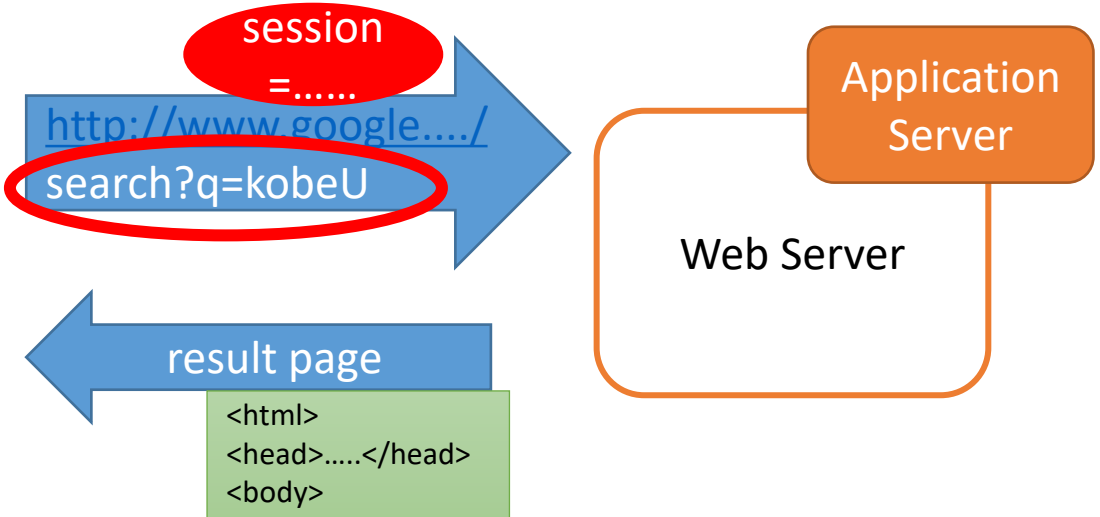
そのまえに、そのまえに、
ネットワーク接続

- ホスト名: ネットワーク機器に割り付けた名前
  - Fully Qualified Domain Name(完全修飾ドメイン名)
  - 例: www.kobe-u.ac.jp
  - DNS サービスが IP address (例 133.30.30.70)に変換
- ポート番号:プロセス／サービス識別用の番号
  - http: 80, https: 443, …



Web Application

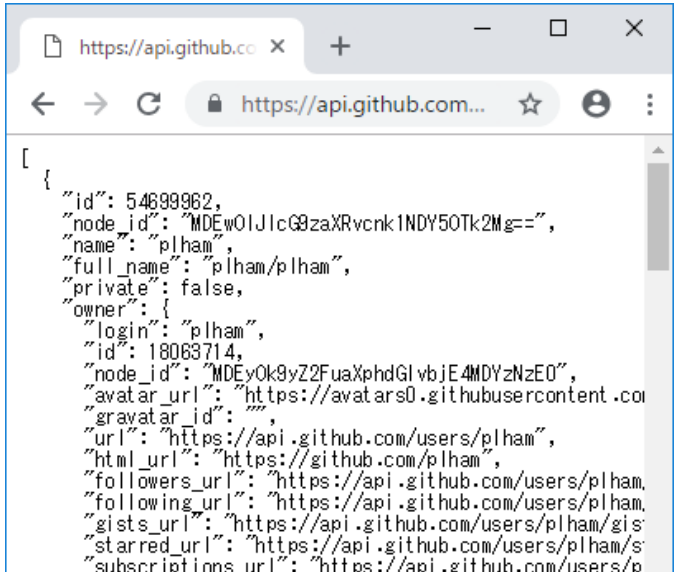
- Web Application: ブラウザ上で動作するアプリ
  - 例： web mail, online shop, …
- 検索エンジンの例
  - URL 中にパラメータを指定可能
    - 例: https://www.google.com/search?q=kobeU
  - Cookie: ブラウザが Server に送信。主にsession識別用。



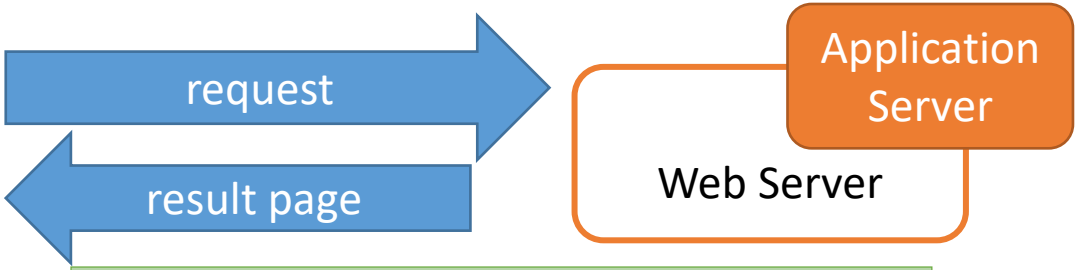
Web Service

各種 クラウドサービスの中でも、
データベースや認証など、
Web Service 経由で操作可能なもの多い

- Web Service:
  (広義)Web 上で提供される情報サービス
  - 機械・プログラム処理用に規格化(XML, JSONなど)
  - Web API として対外公開されているものも多い
  - 以下は github の例 (授業中に別の例もピックアップ予定)



https://api.github.com/users/plham/repos [link]



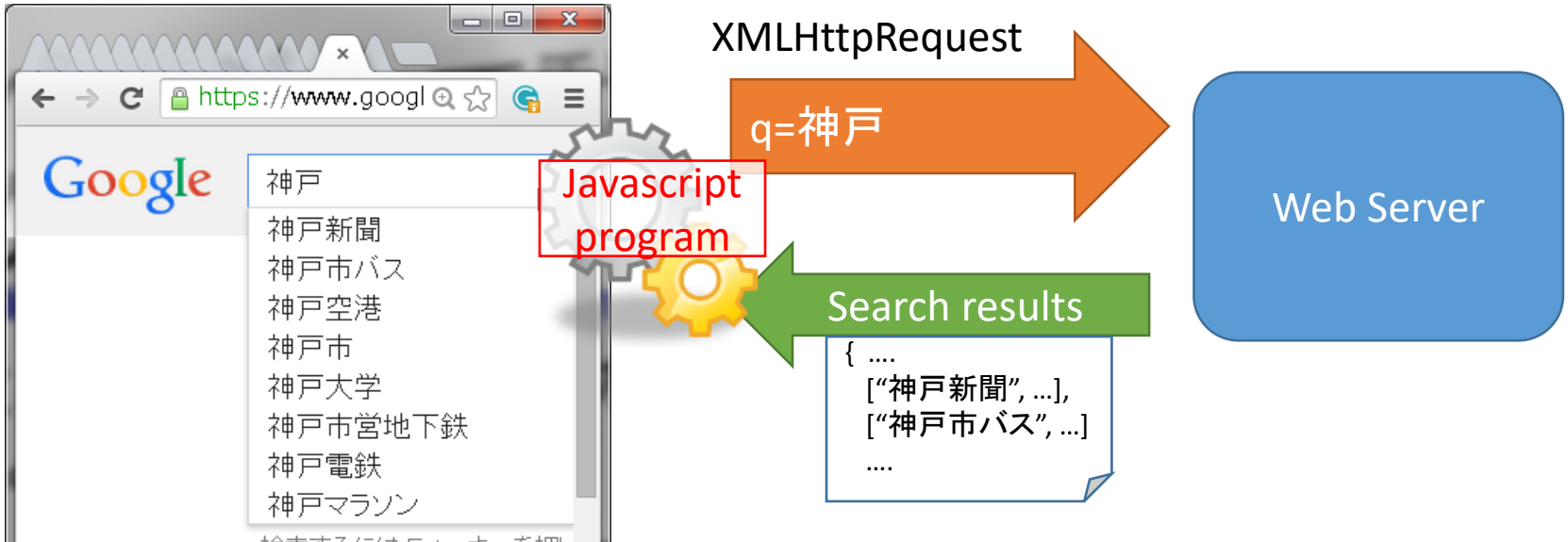

```
[
 {
  "id": 54699962,
  "node_id": "MDEwOlJlcG9zaXRvcnk1NDY5OTk2Mg==",
  "name": "plham",
  "full_name": "plham/plham",
  "private": false,
```

動的 Web ページ

- キー入力に反応するWeb ページとかってありますよね？
  - 例：google suggest, google map

動的 Web ページ

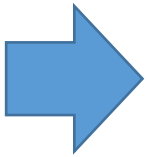
- JavaScript program を利用
  - Event 処理も可能
- AJAX (Asynchronous JavaScript + XML)
  - XMLHttpRequest を発行
  - 結果はCallBack関数で処理



動的 Web ページ

- Javascript の操作対象：HTML文書相当 (DOM)
  - GUI コンポーネント相当が木構造になっているだけ
  - E.g. document.getElementById(“**id**”).innerHTML=“hoge”

```
<html>
<head> …</head>
<body>
  ….
  <div id=“id”>
      ……
  </div>
 …..
 ……
```



```
<html>
<head> …</head>
<body>
  ….
  <div id=“id”>
     hoge
  </div>
 …..
```

単に、ブラウザ上の DOM データを操作しただけ。
サーバデータを操作したわけではない。

Demonstration

今回の授業では何教えるの？

- JavaScript Web アプリではなく、
  Java native アプリを想定
  - DOM 操作の代わりに Swing or Android を利用
  - 非同期処理などの原理は、基本同じ

- サーバサイド「プログラム」は教えない
  1. 既存の Web サービスを利用
  2. クラウド上のストレージサービスを利用

技術トピック

- 遠隔問い合わせ
  - 非同期処理・コールバック処理
  - 構造化文書の処理 (JSON, XML)
- マルチスレッド処理
  - サーバ:
    複数クライアントへの対応
  - クライアント:
    複数の作業を並行処理
- 各種 I/O の安全な処理