

## フレームワーク基礎実習

# 01. Webシステム入門

株式会社ジードライブ

# 今回学ぶこと

---

- Webシステムについて
  - 静的なWebページと動的なWebページ
- フレームワークの概要

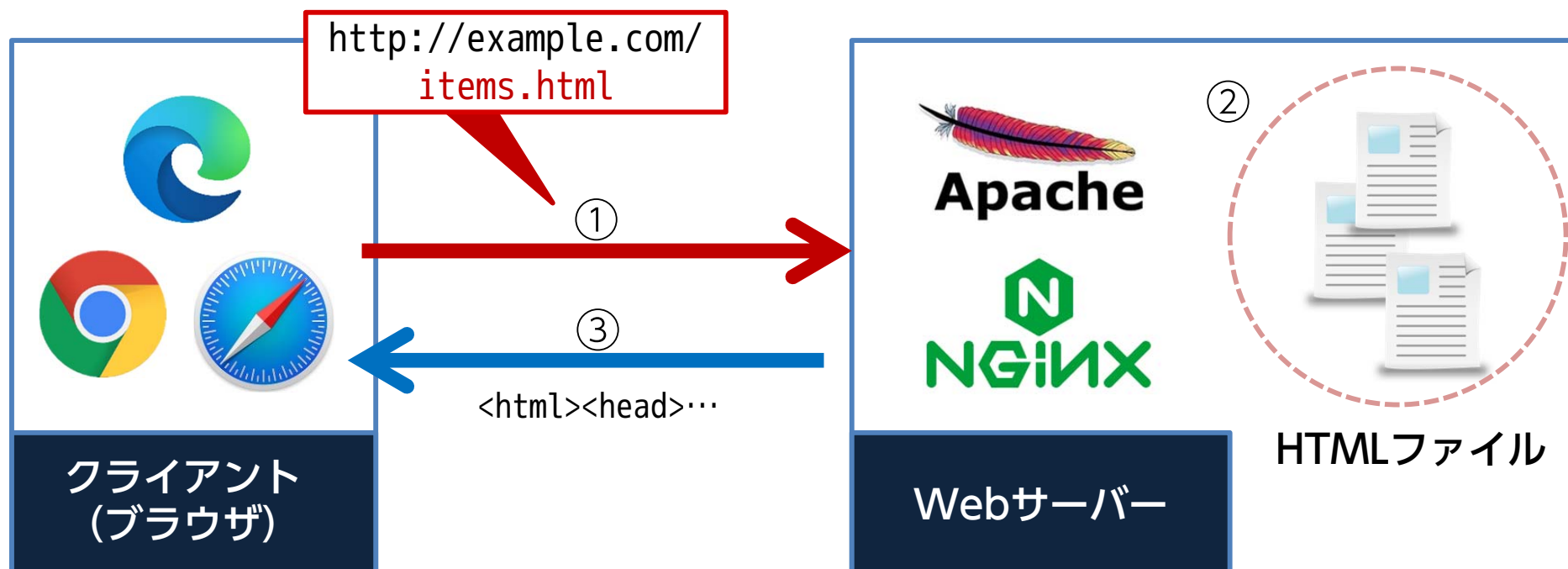
# Webシステムとは

---

- Webサーバー、WebブラウザなどWeb関連技術を中心に構築されたシステム
  - ショッピングサイト
  - SNSサイト
  - ブログサイト
  - 顧客管理システム
  - コンテンツ管理システム
  - 施設予約システム …など
- HTML, CSSに加えて、JavaやPHPなどのプログラミングが必要になる
  - JavaやPHPなどがデータベース(DB)との橋渡しとなって、DB内にデータを追加したり、DB内のデータを読み出すことができる  
⇒ 動的なWebページを作ることができる

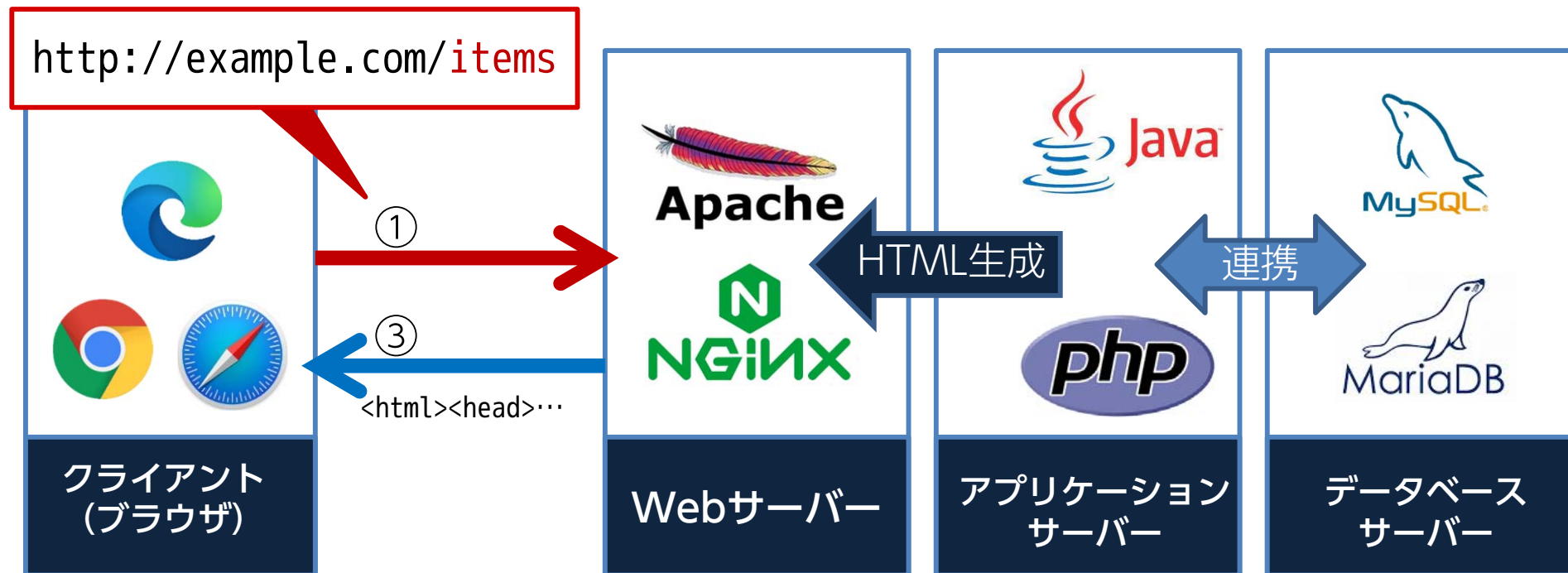
# 静的なWebページ

- ① ブラウザがURLを通じてリクエストを送る
- ② Webサーバーが該当するHTMLファイルを探す
- ③ レスponseとしてHTMLデータを送る



# 動的なWebページ(Webシステム)

- ① ブラウザがURLを通じてリクエストを送る
- ② JavaやPHPなどのプログラムがデータベースと連携し、HTMLのデータを生成する
- ③ レスポンスとしてHTMLデータを送る



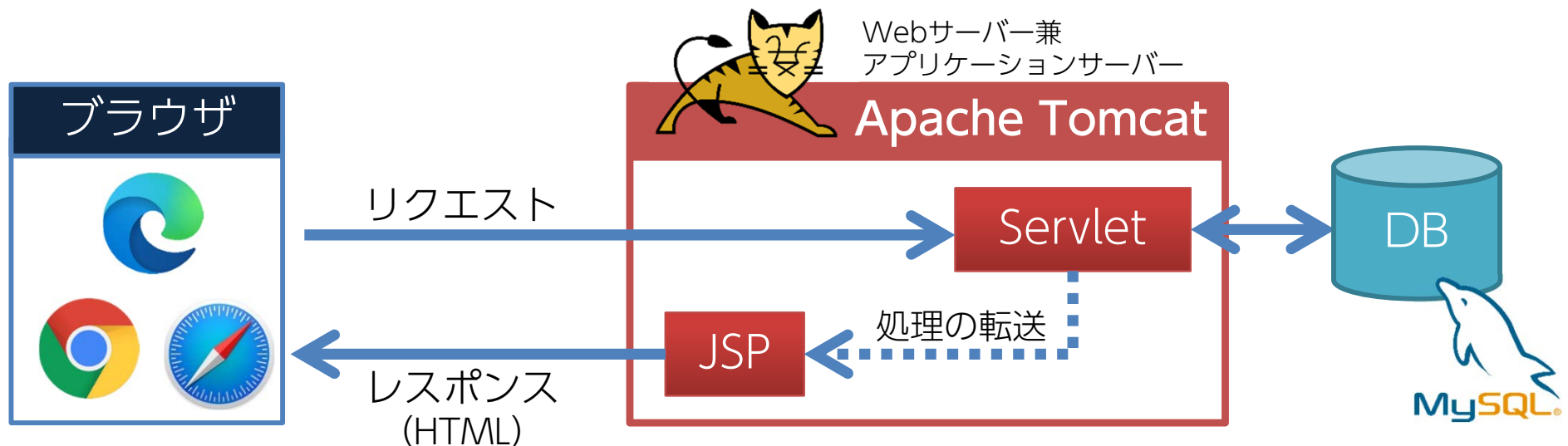
# 静的なWebページと動的なWebページ

---

- 静的なWebページでは、リクエストに対し、あらかじめ用意しておいたHTMLをレスポンスとして返すので、ページが表示されるまでの時間が短い
  - 100個の商品を扱うショッピングサイトで、各商品ごとにページを用意すると仮定すると、あらかじめ100個のHTMLファイルを作成しておく必要がある
- 動的なWebページでは、リクエストが届いてから、HTMLデータを生成するため、静的Webページに比べレスポンスまでの時間が長い
  - 100個の商品に対して、各商品ごとのページを作成する場合、商品のデータをあらかじめデータベースに登録しておく  
⇒ 生成するHTMLデータのひな形となるファイルを1つ用意する

# Webシステムの開発

- JavaによるWebシステムの開発では、Servlet・JSPというサーバー上で動作するJavaの技術を利用する
  - Servletはリクエストを受け、計算やDBとの連携などを行う
  - 画面に表示するデータ(HTML)の生成はJSPが行う
  - ServletやJSPを動作させるにはServletコンテナ(アプリケーションサーバー)が必要 ⇒ Apache Tomcatがこの役割を担う



# Servlet

- HttpServletを継承するJavaのクラスでリクエストを受けて、計算やDBとの連携処理などを行うことができる

例：フォームに入力された商品名と値段を受け取り、DBに登録するServlet

```
@WebServlet("/item/add")
public class AddItemServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request,
                           HttpServletResponse response)
                           throws ServletException, IOException {
        String name = request.getParameter("name");
        int price = Integer.parseInt(request.getParameter("price"));
        Item item = new Item(name, price);
        DaoFactory.createItemDao().insert(item);
        request.setAttribute("item", item);
        request.getRequestDispatcher("/WEB-INF/addDone.jsp")
            .forward(request, response);
    }
}
```

フォーム入力値の取得

DBに商品データを登録

JSPに処理を転送



# JSP

- JSP(Java Server Pages)には、HTMLのコードとJavaのコードを混在させて記述することができる
  - Servletに変換されてから実行される

例：前頁のServletから処理を引き継ぎ、HTMLデータを生成するJSP

```
<%@ page pageEncoding="UTF-8" %>
<%@ page import="java.time.LocalDate" %>
<% LocalDate today = LocalDate.now(); %>
<!DOCTYPE html>
<html lang="ja">
<head>...</head>
<body>
<h1>登録完了</h1>
<p>以下の商品を<%= today %>に登録しました</p>
<p>商品名: ${item.name} <br>値段: ${item.price}円</p>
</body>
```

Javaのコード

Servletから引き継いだデータ

---

# フレームワーク

# フレームワークによる開発

- 現在の開発では、効率化のために、フレームワークが利用されるが多い
  - Java以外の言語で開発を行う場合も、以下のようなフレームワークが使用される

開発言語	代表的なフレームワーク
Java (JSP/Servlet)	Spring Framework
PHP	Laravel
Ruby	Rails
Python	Django
JavaScript	React(フロント) / Express(バックエンド)

# フレームワークとは

---

- フレームワーク = 骨組み、枠組み、型
- プログラムの開発におけるフレームワークとは、プログラムを効率よく開発するためのひな形となるプログラムモジュールを指す
  - モジュール：何かしらの機能をもつ部品  
組み合わせて使用することが多い

# Webシステムのフレームワーク

---

- Webシステム開発ではどのシステムでも必要になる共通の機能やしくみが存在する
  - MVCモデル
  - バリデーション
  - データベースアクセス
- フレームワークはこのような共通機能や開発効率およびメンテナンス性の高いアプリケーション構造の「枠組み」を提供する

# ライブラリとフレームワーク

- ライブラリ
  - よく利用する機能を集めたプログラムの部品
  - Javaの場合はクラスの集合であるクラスライブラリ(JARファイル)として提供されている
  - 自身でアプリケーション全体の実装を行い、その中で必要なライブラリを呼び出して利用する



# ライブラリとフレームワーク

- フレームワークソフトウェアで良く使われる機能があらかじめ用意された枠組み(ひな形)
  - ライブラリの集まり
  - フレームワークを土台として、アプリケーション固有の機能を自身で実装し、追加していく



# フレームワーク導入のメリット・デメリット

---

## メリット

- アプリケーションに必要な機能の作り込みが出来上がっているため、開発工数の削減やアプリケーションの品質の向上などが期待できる

## デメリット

- フレームワーク固有の概念や個々のフレームワークの使い方などを習得するための学習コストが発生する



# Javaのフレームワーク

製品名	概要
Spring Framework <b>Spring Boot</b>	効率的なアプリケーション開発を行うために必要な機能を包括的に含むフレームワーク Spring BootはSpring Frameworkを発展させ、使いやすくしたもの
<b>MyBatis</b>	データベース利用時に導入するフレームワークで、メソッドとSQL文のマッピングを行うことができる
Play Framework	Ruby on Rails や Django(Python)の影響を受けたフレームワークで、Java及びScalaに対応している

訓練では、Spring BootとMyBatisを学習する

# Spring BootとWebシステム

- Spring BootとMyBatisを導入したWebシステムの動作イメージは以下ようになる
  - Spring Bootが内包するApache Tomcat上でServletがリクエストを受け取り、Controllerと呼ばれるオブジェクトを呼び出す
  - ControllerはMyBatisを使いDBと連携し、画面に表示するデータの生成はテンプレートエンジンであるThymeleafが担う

