最速でJavaエンジニアになって稼ぐ

2018/11/01

人生逃げ切りオンラインサロン内

目次的なやつ

- そもそもWebアプリケーションって何。
- Webアプリケーションでよく使う(見かける)言葉。
- JavaのWebアプリケーション開発の流れ。
- Servlet/JSP
- アプリケーションサーバー
- とにもかくにもWebアプリ動かす

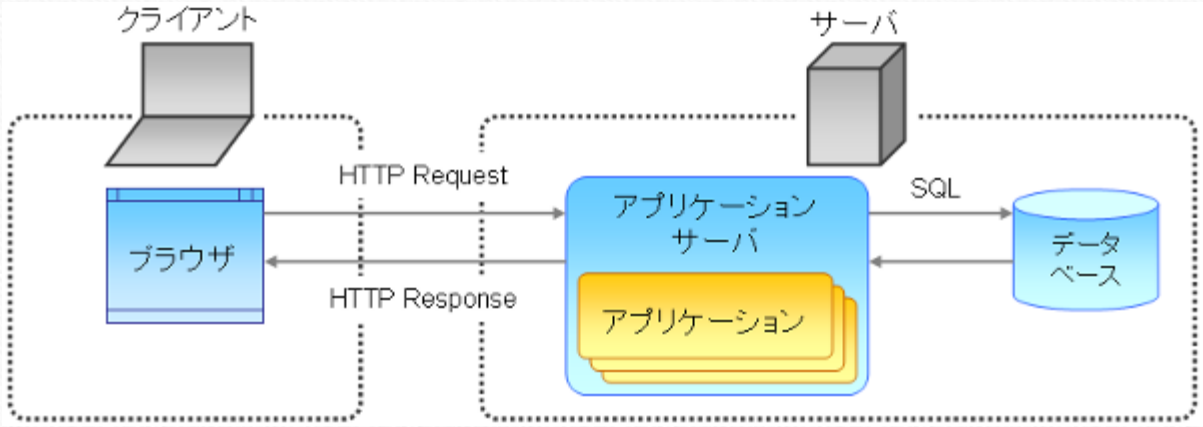
そもそもWebアプリケーションって何。

- 端的に言えば「ブラウザ上で利用できるアプリケーション」の事。
- クライアント(利用者)とサーバー(提供者)の関係性があり、クライアントの要求(リクエスト)に対してサーバーが応える(レスポンス)。
- 基本的にリクエストに対するレスポンスはブラウザ上での画面表示になる。リクエストはほぼ文字情報となるので、それを元にサーバー側が処理した結果を返す、という流れ。
- その為、求められる技術としては「ブラウザの仕様(挙動)」「通信」「サーバー側の仕様(挙動)」「プログラミング(Javaとか)」「HTMLやCSSなどの画面描画」がある。
- 上記を見てわかるようにJavaとかPHPだけ、では足りずHTML/CSSや通信、ブラウザの挙動とかも理解してないとアプリケーションは開発できない。

Webアプリケーションでよく使う(見かける)言葉。

クライアントとサーバー

- 昔からよく「クラサバ構成」とか言われたやつだけど、最近はブラウザ上で動くWebアプリを指すことが多い。
- 別にブラウザ限定ではく利用者と提供者の関係が出来てたら良い。
- Webアプリとしては下記のようなイメージ(貰い物



リクエストとレスポンス

- リクエスト自体はサーバーなどの提供者に対して送るものなので、Webアプリ限定ではない。
- 最近よくあるのは「API」と呼ばれるもの。これもリクエストを送る、という意味で合うので使う。
- Webアプリケーションという前提ありきの場合は「HTTP Request」を指す。
- レスポンスも同様。Webアプリ前提の場合は「HTTP Response」を指す。
- ちなみにここらへんのHTTPほにゃららはブラウザがやってくれてる。
- ブラウザを介してHTTP Requestを送り、サーバーがHTTP Responseを返してブラウザに表示する。そんな感じ。

通信

- ブラウザからWebアプリケーションを動かす場合は「URL」が必要になる。
- これはサーバー(提供者)がどこにいるかを指し示すもので「URI」の一部。
- ブラウザURLを入れる事もれっきとしたリクエストになるので、それに対してサーバーはレスポンスを返す。だから我々はブラウザ経由で画面を見れている。世の中はWebアプリケーションだらけって事。
- こうしたリクエストとレスポンスのやり取りを「通信」と言う。
- 通信は一件単純に見えるけど、実際には「どういった内容」を「どのような形」にして「どのようにして送るか」みたいな事をガッチリ決めてやっているので、通信フォーマットと言われるものがいくつかある。
- それが「JSON」とか「XML」とかだったりする。
- この取り決めがないと受け取る側は、毎回独自形式の情報を自前で判断して処理する必要が出るので、そんなん無理ゲーになっちゃう。

HTTPメソッド

- 主に「GET」と「POST」。他にもあるけどあんまり使わないので割愛。
- GETは名の通り、取得する。さっきの通信のやり方の一部でJSONとかの形式が決まってたとしても、どうやってやり取りするかはわからない。それがHTTPメソッド。GETは取得のみに利用する。
- POSTも一応名の通りだけどイメージはしにくい。これは情報を渡して、アプリケーション側で登録したり、更新したりといった処理を行うときに使う。
- 厳密にはDELETEとかのメソッドもあるのでPOSTで全部やるわけじゃないけど、まぁ使い分けしてるのは珍しいってレベル。
- 大きな違いとしてはGETは「クエリパラメータ」と言ってURLの後に情報が付く。「http://hogehoge.com/?hoge=fuga」の「?以降」がそれ。
- POSTの場合はそうではなく「リクエストボディ」に付く。あまり気にしなくてOK。

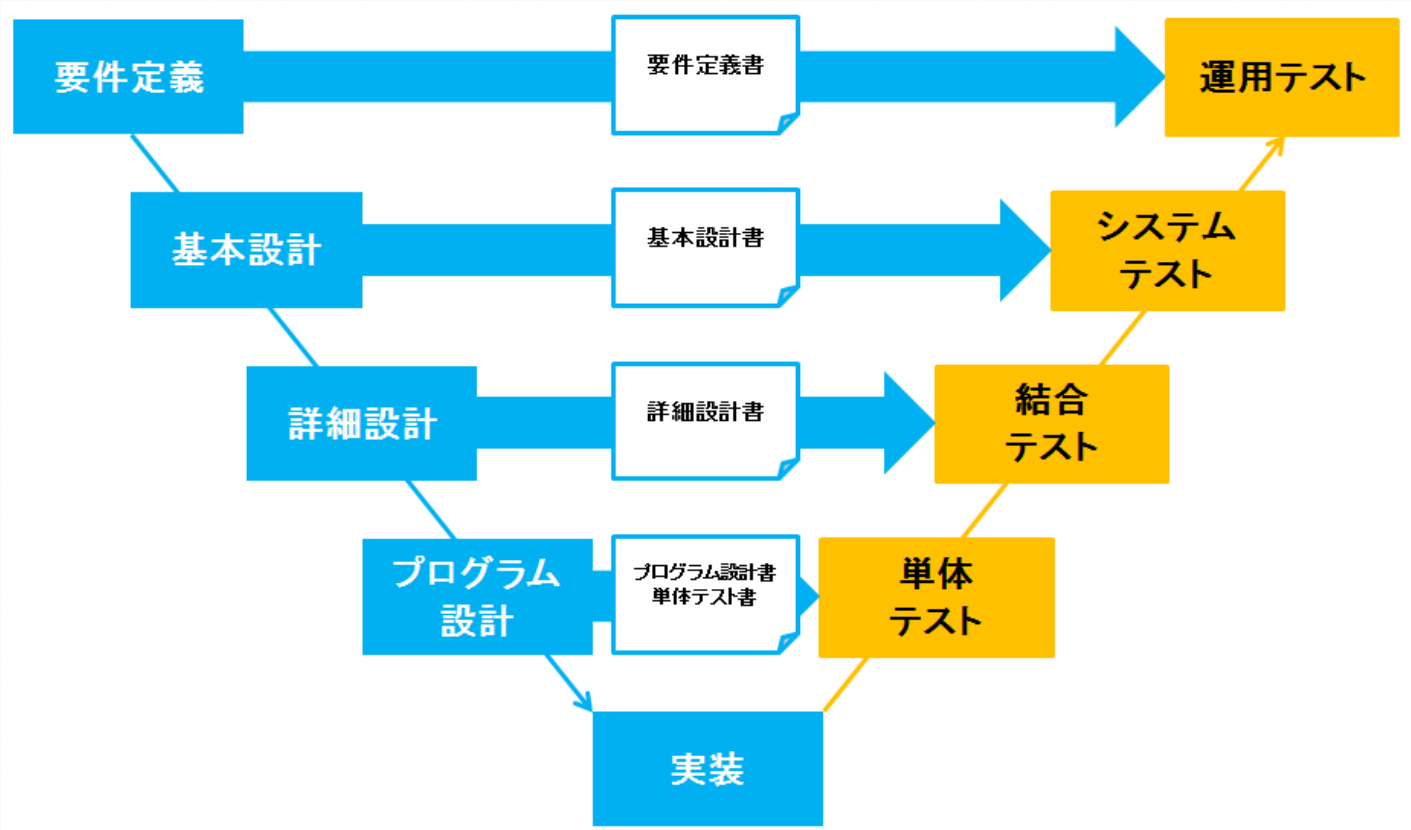
HTMLとかCSSとかJSとか

- HTMLはブラウザで画面描画を行うときに使う「言語」。なので、プログラミング言語と言えなくもないけど、実際にはifとかforとかみたいな要素が全くなくてただ「タグ」と呼ばれるもので表示する内容を定義していくだけなので、プログラミング言語としては扱われない。
- CSSはHTMLで書かれたものをスタイリングするもの。スタイルシートとも言う。文字色変えたり、フォント変えたりというのはこのCSSで行う。
- JSはプログラミング言語。JavaScriptが正式名称だけど、Javaとは無関係なので注意。動的型付けでしかもブラウザ利用を前提としているので、ブラウザ上でのアニメーションとかに使われる。
- HTML/CSS/JSもそれぞれバージョンやフレームワークがあったりする。
- 今はHTML5とCSS3でJSは「jQuery」「Angular」「React」「Vue」とかある。

JavaのWebアプリケーション開発の流れ。

Webだろうとそうじゃなかろうと使う技術が変わるだけで、流れは実はあんまり変わらない。

- ぶっちゃけ言語が変わってもあまり変わらない。良く言われるのが「V字モデル」。他にも色々なモデルはあるけど、まずは押さえておきたいモデル。



- アプリケーションを作る時はまず「要件(して欲しい事)」があって、それを実現するための作り方や体制、仕組みとかを色々決める。「要件定義」「アーキテクチャ選定」とか色々ある。
- そこからやっと作るものが決まってきて「機能」の数とか種類がわかってきて、それをどう作るかという具体的なところに向かう。これが「設計」。
- 規模が大きい場合は、色々な人が同時に作る事になるので「全体設計」とか言って、設計の仕方とか作り方を細かく取り決めたりする。めっちゃ時間かかる。考えなきゃいけない事多すぎて。仕方ないけど、面倒。費用勿体ない。ジレンマ。
- 設計が出来てから実装。後は出来上がったものが求めていたものになっているかをテストして、納品して、完了。簡単でしょ。

閑話休題

ちゃんとやってりゃ上手くいくのに、何故失敗するのか

- 実は結構事情は複雑。極端な話をするなら「全員」が「スケジュール通り」に「作業」を「完了」させれば何も問題は起きない。
- そんなん出来るか。
- 全てを予見できるわけないから、何かが起きた時に何かが犠牲になる。その対処をどうするか、という話。
- 「問題は起きる」「それなら問題を早く起こして、早く対処すればいい」というような考え方もある。問題を先延ばしにしない。失敗を評価する、そんな感じ。
- ここらへんは開発手法として、纏めて話すつもりだけど「開発自体をどのようなプロセスで行うか」というのも一つの技術なので、考えておいて欲しい。
- 「時間が無制限」という前提に立つならば、誰でも出来るんだから技術者もいらない。技術者とは何なのか。振り返ってみよう。

話は戻って

Servlet/JSP

- リクエストに対する処理を行うのがJavaにおいては「Servlet」になる。
- 他にもServletに行く前に「Filter」とかもあるけど割愛。
- Servletでリクエストで受け取った内容を元に何らかの処理をしてレスポンスを返す。
- でもこの時に処理した内容に基づいて「動的に結果を変えて画面表示したい」。変数の内容を表示したり、みたいな。
- でもHTMLはタグで定義するだけだから変数を受け取ってどうこうは出来ない。じゃぁどうする。
- 「JSP」を使う。Javaで編集したレスポンスをJSPで受け取る事で動的に扱える。中身はHTMLとほぼ一緒だけど、HTMLに新機能を追加しているようなイメージ。

アプリケーションサーバー

- Servletで処理した内容をJSPで表示する。
- じゃあそのServletとJSPはどうやって作るのか、どうやって動かすのか。
- 作るのは今までと同じようにJavaプログラムとして作る。
- 動かすには「ビルド」をして「war」にして「アプリケーションサーバー」に配備する。
- アプリケーションサーバーというのはJavaアプリケーションを動かす為のサーバー。Webサーバーとプログラムを実行するJVMを搭載したサーバーという認識でOK。
- 補足としてJavaにはJavaEEというエディションがあり、これはエンタープライズエディションの略になる。そして、アプリケーションサーバーはJavaEEが搭載されているので、アプリが動く。
- 実はJavaSEは開発用なので、サーバー上で動かすには色々と足りないものがあったりする。その足りないものはJavaEEに入っている。
- またまた捕捉で注意事項として、Java11からはJavaEEに関するものが一部、JavaSEから削除されているので、以前使えたものが使えなくなったりしている。

とにもかくにもWebアプリ動かす

アプリケーションサーバーのインストール

Webサイトを参考にまずはApache
Tomcatのインストールから。
バージョンは8以降であれば恐らく大丈夫。

そして相も変わらずネットの海から既にあるものは全力で利用するスタイル

参考

http://www.javaroad.jp/opensource/js_tomcat1.htm

Eclipse上でTomcatをいじれるようにプラグインを入れてみる

参考

https://qiita.com/sabineko/items/e21aaaf05706e1ec0acb

適当なアプリを作って、動かしてみる

参考

http://www.javaroad.jp/opensource/js_eclipse6.htm

次回までの宿題

出来なかった人は出来るまで。
出来た人はCRUD処理を実装してみよう。

- CRUD処理については調べる。すぐに分かる。
- JavaプログラムからDBへの接続が必要になる。
- 今後はMySQLの5.7以降を使う予定。なのでMySQLと繋ぐ必要有。
- JDBCコネクタというものが必要。これはDBと繋ぐためのモジュール。
- JDBCコネクタはダウンロードしてくる必要がある。ライブラリとして追加しないといけない。
- 最終的にはアプリケーションサーバーと繋いで動かす事になるけれど、仕組みの理解として、まずは普通にJavaプログラムから接続でOK。
- MySQLをインストールした上でDBとテーブル作る必要がある。

参考

https://www.task-notes.com/entry/20150414/1428980400

おしまい