Google App Engine(Java)

・https://cloud.google.com/appengine/docs/standard/java/building-app/environment-setup?hl=JA
・個人的に最強の PaaS プラットフォーム Google App Engine の良い所を挙げてみる

・https://cloud.google.com/appengine/docs/standard/java/tools/uploadinganapp?hl=JA

■ Maven のダウンロードとインストール

・http://maven.apache.org/

■ Maven プロジェクト作成

・https://cloud.google.com/appengine/docs/standard/java/tools/using-maven?hl=JA
・appengine-skeleton-archetype を使用

```
mvn archetype:generate -Dappengine-version=1.9.59 -Dapplication-id=[YOUR-PROJECT-ID]
-Dfilter=com.google.appengine.archetypes:
```

■デプロイ

・pom.xml のディレクトリで

```
> mvn appengine:deploy
```

・デプロイ後、ブラウザで開く

```
> gcloud app browse
```

静的コンテンツ

・https://cloud.google.com/appengine/docs/standard/java/building-app/static-content?hl=JA

■静的ファイルの配置場所

・webapp ディレクトリ内に置きます。フォルダも使用できますが、すべてのファイルパスと URI は webapp ディレクトリからの相対パスになります
・静的ファイルの場所を選択したら、その場所を appengine-web.xml ファイル内の <static-files> 要素で定義する必要があります。

フォームデータの処理

・https://cloud.google.com/appengine/docs/standard/java/building-app/handling-form-data?hl=JA

■入力ページへのリンク

```
<a href='/jsp/form'>Form</a></td>
```

■入力ページ Jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>HTTP Form</title>
</head>
<body>
  <form method="POST" action="/form/save">
    <label for="save_content">内容 :</label>
    <textarea name="save_content" rows="3" cols="50">保存内容 </textarea>
    <button type="submit">保存 </button>
  </form>
</body>
</html>
```

■ JSP 用ディスパッチャー

・JSP を WEB-INF 配下においてリダイレクトする

```
@WebServlet(
    name = "JspDispatcher",
    urlPatterns = {"/jsp/*"}
)
public class JspDispatcher extends HttpServlet{
  private static final String JSP_PATH_PATTERN = "/WEB-INF%s.jsp";

  @Override
   protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {
    req.getRequestDispatcher(
        String.format(JSP_PATH_PATTERN, req.getRequestURI())).forward(req, resp);
  }
}
```

■主処理

```
@WebServlet(
    name = "FormController",
    urlPatterns = {"/form/*"}
)
public class FormController extends HttpServlet {
  private static final long serialVersionUID = 1L;

  @Override
   protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {
    req.setAttribute("content", req.getParameter("save_content"));
    req.getRequestDispatcher("/WEB-INF/jsp/result.jsp").forward(req, resp);
  }
}
```

■結果表示

・JSP の escapeXml 機能を使用して、クロスサイト スクリプティング（XSS）攻撃に対する対抗措置をとる

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
```

```
</head>
<body>
  <p>${fn:escapeXml(content)}</p>
</body>
</html>
```

Spring Boot

- https://tosi-tech.net/2018/08/spring-boot-on-google-app-engine-standard/
- https://qiita.com/tora470/items/1695a8614551b7500c2a

Google App Engine Java Standard 環境で手っ取り早く Spring Boot アプリケーションを開発する

Cloud Data Store

- https://cloud.google.com/appengine/docs/standard/java/building-app/cloud-datastore?hl=JA