**2.Servlet**

서블렛은 간단히

서블릿 작성과정

javax.servlet.http.HttpServlet 클래스로부터 상속받아서 작성

서블릿 규칙에 따라 자바 로 작성

경우에 따라서 web.xml파일에서 서블릿을 주소와 맵핑

\*맵핑이란

서블릿 요청 처리

요청 방식에 따라 doget dopost

1. 클라이언트측에서 WAS로 요청하는 방식이 두가지 : Get방식, Post방식이 있다.

(HTML코드안에 전송하는 방식 지정해 주는 태그에 인해 지정 태그에 get으로 할지 post로 할지에 따라 정해짐)

 Get방식 – URL값으로 정보가 전송되어 보안에 약함

 주소와 매개변수를 붙여서 주소 표시줄에 입력하는 방법(?로 구분)

 a 태그를 이용해서 페이지를 요청하는 경우

 form태그에서 명시적으로 GET방식으로 요청하는 경우(method=”get”)

 매개변수의 데이터는 255자 이내이며 보안성이 없음

 Post방식 – header를 이용해 정보가 전송되어 보안에 강함

 form태그에서 명시적으로 POST 방식으로 요청(method=”post”)

 데이터의 크기에 제한이 없으며 URL에 표시되지 않으므로 보안성이 우수

 한글 파라미터 전달시 request.setCharacterEncoding(“utf-8”)추가

Service()도 요청 받음

request객체로 요청 데이터를 읽어오고 response 로 응답하여 데이터 전송할수있다.

컨텍스트 패스(Context Path)

서블렛 작동순서

웹브라우저 → [웹서버] → [웹어플리케이션서버] → [Servlet 컨테이너 servlet객체생성, 쓰레드생성]

서블릿 컨테이너

우리가 서버에 서블릿을 만들었다고 해서 스스로 작동하는 것이 아니고 서블릿을 관리해주는 것이 필요한데, 그러한 역할을 하는 것이 바로 서블릿 컨테이너 입니다. 예를 들어, 서블릿이 어떠한 역할을 수행하는 정의서라고 보면, 서블릿 컨테이너는 그 정의서를 보고 수행한다고 볼 수 있습니다. 서블릿 컨테이너는 클라이언트의 요청(Request)을 받아주고 응답(Response)할 수 있게, 웹서버와 소켓으로 통신하며 대표적인 예로 톰캣(Tomcat)이 있습니다. 톰캣은 실제로 웹 서버와 통신하여 JSP(자바 서버 페이지)와 Servlet이 작동하는 환경을 제공해줍니다.

1. 웹서버와의 통신 지원서블릿 컨테이너는 서블릿과 웹서버가 손쉽게 통신할 수 있게 해줍니다. 일반적으로 우리는 소켓을 만들고 listen, accept 등을 해야하지만 서블릿 컨테이너는 이러한 기능을 API로 제공하여 복잡한 과정을 생략할 수 있게 해줍니다.그래서 개발자가 서블릿에 구현해야 할 비지니스 로직에 대해서만 초점을 두게끔 도와줍니다.

2. 서블릿 생명주기(Life Cycle) 관리 서블릿 컨테이너는 서블릿의 탄생과 죽음을 관리합니다. 서블릿 클래스를 로딩하여 인스턴스화하고, 초기화 메소드를 호출하고, 요청이 들어오면 적절한 서블릿 메소드를 호출합니다. 또한 서블릿이 생명을 다 한 순간에는 적절하게 Garbage Collection(가비지 컬렉션)을 진행하여 편의를 제공합니다.

3. 멀티쓰레드 지원 및 관리 서블릿 컨테이너는 요청이 올 때 마다 새로운 자바 쓰레드를 하나 생성하는데, HTTP 서비스 메소드를실행하고 나면, 쓰레드는 자동으로 죽게됩니다. 원래는 쓰레드를 관리해야 하지만 서버가 다중 쓰레드를생성 및 운영해주니 쓰레드의 안정성에 대해서 걱정하지 않아도 됩니다.

4. 선언적인 보안 관리 서블릿 컨테이너를 사용하면 개발자는 보안에 관련된 내용을 서블릿 또는 자바 클래스에 구현해 놓지 않아도 됩니다.일반적으로 보안관리는 XML 배포 서술자에 다가 기록하므로, 보안에 대해 수정할 일이 생겨도 자바 소스 코드를 수정하여 다시 컴파일 하지 않아도 보안관리가 가능합니다.

asdasda

sdasdasdasdasdasdas

dasdasdasdasdasdasdasdasdasdasd

[ Servlet 생명주기 ]

최초 한번

Servlet 객체 생성

선처리 @PostConstruct

init() 호출

최초 한번

service() doGet() doPost() 호출

요청시 매번

destroy() 호출

마지막 한번(자원해제: servlet수정, 서버 재가동 등등..)

후처리 @PreDestroy

서버종료및재가동

/애플리케이션 종료

클라이언트의 요청이 들어오면 컨테이너는 해당 서블릿이 메모리에 있는지 확인하고, 없을 경우 init()메소드를 호출하여 적재합니다. init()메소드는 처음 한번만 실행되기 때문에, 서블릿의 쓰레드에서 공통적으로 사용해야하는 것이 있다면 오버라이딩하여 구현하면 됩니다. 실행 중 서블릿이 변경될 경우, 기존 서블릿을 파괴하고 init()을 통해 새로운 내용을 다시 메모리에 적재합니다.

init()이 호출된 후 클라이언트의 요청에 따라서 service()메소드를 통해 요청에 대한 응답이 doGet()가 doPost()로 분기됩니다. 이때 서블릿 컨테이너가 클라이언트의 요청이 오면 가장 먼저 처리하는 과정으로 생성된 HttpServletRequest, HttpServletResponse에 의해 request와 response객체가 제공됩니다.

컨테이너가 서블릿에 종료 요청을 하면 destroy()메소드가 호출되는데 마찬가지로 한번만 실행되며, 종료시에 처리해야하는 작업들은 destroy()메소드를 오버라이딩하여 구현하면 됩니다.