Generic REST Endpoint mit Spring Boot

Die Programmierung von REST Services mit Spring Boot ist gemäss Lehrbuch eine relativ einfache Sache.Jede Klasse kann als Rest Service funktionieren und Daten im Format JSON verarbeiten. In der Regel nimmt im Lauf der Entwicklung die Anzahl REST Services zu und damit auch die Komplexität und Redundanz. So fehlt ein zentrales Error Handling oder ein Überwachungspunkt (Single Rest Endpoint). Ein Single Rest Endpoint arbeitet wie ein Portier, alle Zu- und Abgänge werden über einen Punkt abgewickelt. Er zwingt uns ein Protokoll für die Kommunikation zu definieren und damit die Rest Endpoints zu standardisieren. Genau hier hilft das Konzept des generischen Standalone Rest Endpoints. Die Klasse GenericRestService zeigt anhand der Methode Get den Servlet Endpoint. Dieser Call behandelt allfällige Exceptions und leitet den Call an die Methode executeGet weiter.

```
package ch.std.service.generic;&#xA;import javax.servlet.http.HttpServletRequest;&#xA;&#xA;import org.json.JSONObject;&#xA;import org.springframework.http.MediaType;&#xA;import org.springframework.web.bind.annotation.RequestMapping;&#xA;import org.springframework.web.bind.annotation.RequestMethod;&#xA;import org.springframework.web.bind.annotation.RestController;&#xA;&#xA;@RestController&#xA;@RequestMapping(value = &#34;/rest/generic&#34;, produces = MediaType.APPLICATION_JSON_VALUE)&#xA;public class GenericRestService {&#xA;&#xA; public GenericRestService() {&#xA;  }&#xA;&#xA;  @RequestMapping(method = RequestMethod.GET)&#xA;  public Object get(HttpServletRequest httpServletRequest) {&#xA; try {&#xA;  JSONObject jsonResponse = executeGet(httpServletRequest);&#xA;  return jsonResponse;&#xA;  } catch(Exception e) {&#xA;  JSONObject jsonObject = new JSONObject();&#xA;  try {&#xA;  jsonObject.put(&#34;status&#34;, &#34;failure&#34;);&#xA;  } catch (Exception ee) {&#xA;  }&#xA;  return jsonObject;&#xA;  }&#xA;  }&#xA;&#xA;  public JSONObject executeGet(HttpServletRequest httpServletRequest) {&#xA;  throw new UnsupportedOperationException();&#xA;  }&#xA;}&#xA;
```

Der effektive Endpoint wird über die Subklasse z.B. EchoRestService implementiert:

```
&#xA;package ch.std.service.echo;&#xA;&#xA;import java.util.Map;&#xA;&#xA;import javax.servlet.http.HttpServletRequest;&#xA;&#xA;import org.json.JSONObject;&#xA;import org.springframework.http.MediaType;&#xA;import org.springframework.web.bind.annotation.RequestMapping;&#xA;import org.springframework.web.bind.annotation.RestController;&#xA;&#xA;import ch.std.service.generic.GenericRestService;&#xA;&#xA;@RestController&#xA;@RequestMapping(value = &#34;/rest/echo&#34;, produces = MediaType.APPLICATION_JSON_VALUE)&#xA;public class EchoRestService extends GenericRestService {&#xA;&#xA;  public EchoRestService() {&#xA;  }&#xA;&#xA;  public JSONObject executeGet(HttpServletRequest httpServletRequest) {&#xA;  Map&lt;?,?&gt; parameterMap = httpServletRequest.getParameterMap();&#xA; JSONObject response = new JSONObject(parameterMap);&#xA;  return response;&#xA; }&#xA;}&#xA;
```

Die restlichen HTTP Methoden POST, PUT und DELETE sind nun einfach integrierbar. Die Klasse HttpServletRequest kann nach Belieben selber gekapselt werden.

Feedback

War dieser Blog für Sie wertvoll. Wir danken für jede Anregung und Feedback

Kontakt

Simtech AG

Finkenweg 23
3110 Münsingen
Schweiz

Impressum

Das Copyright für sämtliche Inhalte dieser Website liegt bei Simtech AG, Schweiz.
Beachten Sie auch unsere Hinweise zum Urheberrecht, Datenschutz und Haftungsauschluss.
Jeder Hinweis auf Fehler nehmen wir gerne entgegen.

Copyright

2024 Simtech AG, All rights reserved, Powered by stack.ch written in Golang by Daniel Schmutz

https://www.simtech-ag.ch/unsupportedoperationexception