Ergänzungen zu Servlets

javax.servlet.annotation

Annotation Type WebServlet

```
@Target(value=TYPE)
@Retention(value=RUNTIME)
@Documented
public @interface WebServlet
```

Annotation used to declare a servlet.

This annotation is processed by the container at deployment time, and the corresponding servlet m

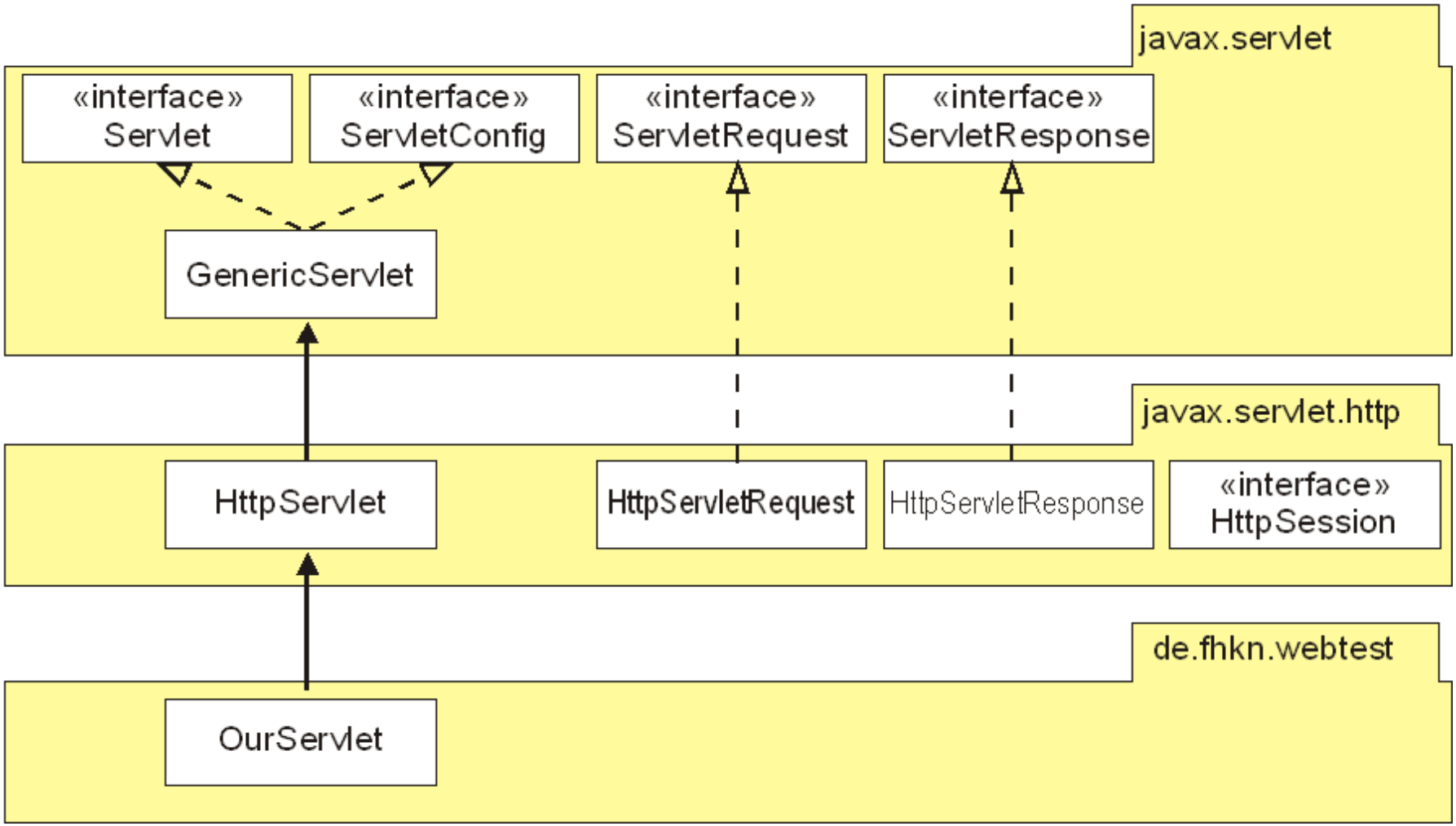
Since:
Servlet 3.0
See Also:
Servlet

Optional Element Summary

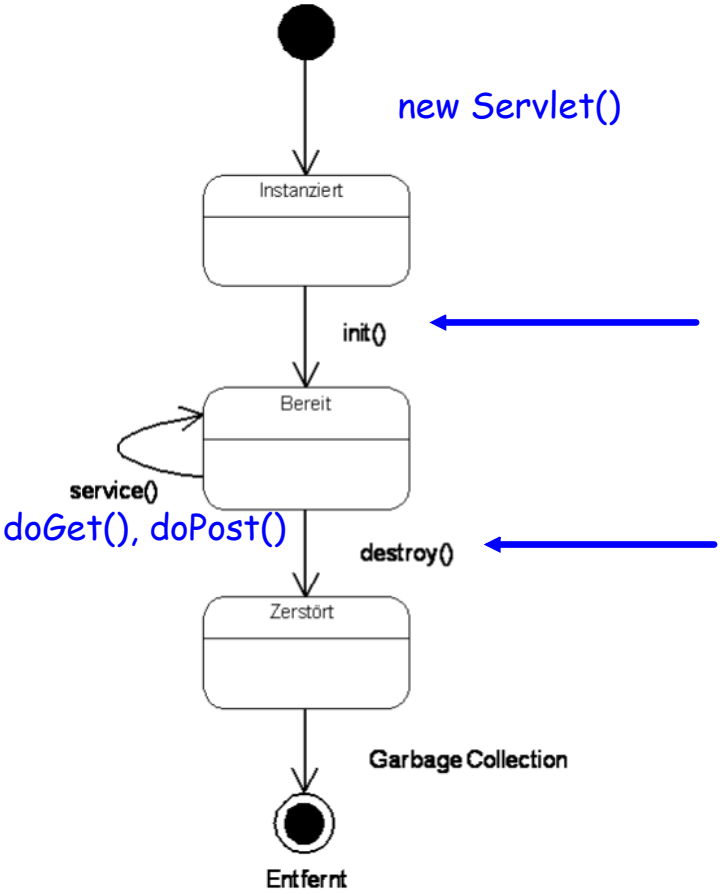
boolean	asyncSupported Declares whether the servlet supports asynchronous opera
java.lang.String	description The description of the servlet
java.lang.String	displayName The display name of the servlet
WebInitParam[]	initParams The init parameters of the servlet
java.lang.String	largeIcon The large-icon of the servlet
int	loadOnStartup The load-on-startup order of the servlet
java.lang.String	name The name of the servlet
java.lang.String	smallIcon The small-icon of the servlet
java.lang.String[]	urlPatterns The URL patterns of the servlet
java.lang.String[]	value The URL patterns of the servlet

```
1  package servlets;
2
3  import java.io.IOException;
16
17 @WebServlet("/ErstesServlet")
18 public class ErstesServlet extends HttpServlet {
19     private static final long serialVersionUID = 1L;
20
21     protected void doGet(HttpServletRequest request,
22         String vname = request.getParameter("vorname"
```

legt fest, welcher Pfad auf welches Servlet geleitet wird (Routing).


javax.servlet
«interface»
Servlet
«interface»
ServletConfig
«interface»
ServletRequest
«interface»
ServletResponse
GenericServlet
javax.servlet.http
HttpServlet
HttpServletRequest
HttpServletResponse
«interface»
HttpSession
de.fhkn.webtest
OurServlet

Servlet Lebenszyklus

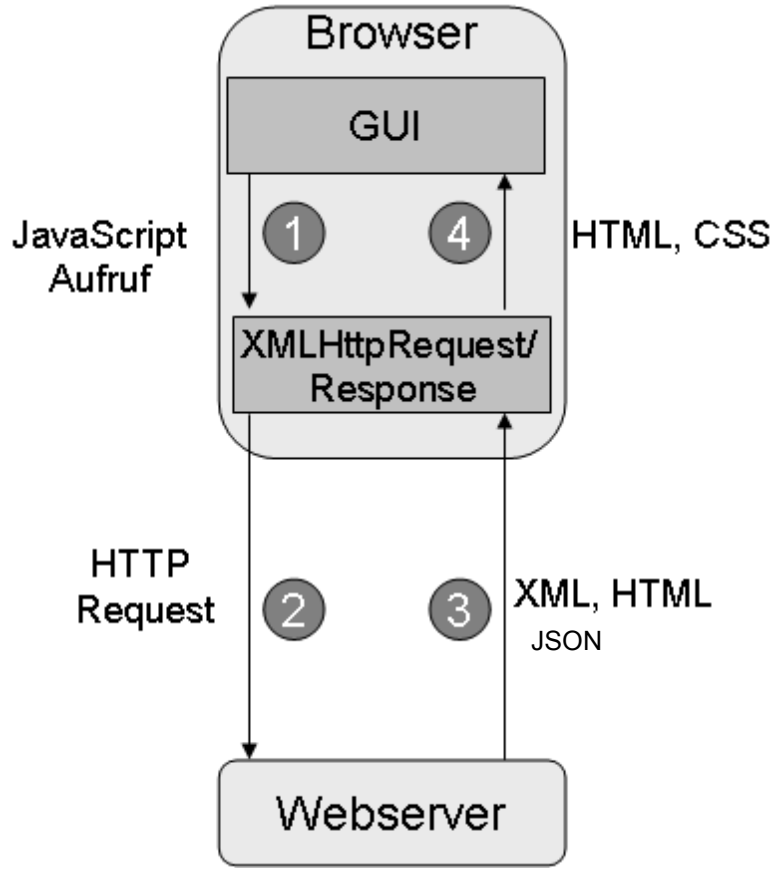


kann, aber muss nicht implementiert werden. Diese Methode wird ebenfalls vom Container aufgerufen und kann beispielsweise genutzt werden, um eine DB-Verbindung aufzubauen.

"Aufräumarbeiten" z.B. DB-Verbindung wieder schließen

AJAX: Asynchronous Java Script and XML

Beispiel: Facebook lädt asynchron (d.h. ohne die komplette Seite neu zu laden) weitere Posts nach, die von JavaScript in die Seite eingebaut werden.



Bewertung von AJAX

+ kein komplettes Laden von Seiten, damit flüssigeres Arbeiten

+ weniger Bandbreite, da nur benötigte Daten nachgeladen werden

- viele Requests

- JavaScript muss aktiviert sein

- verteilte Logik auf Client und Server

- SEO: Teile des HTML-Codes werden erst später geladen und können daher ggf. nicht indiziert werden

```
var url = "URL" + escape(eingabe.value);
var req = new XMLHttpRequest();

req.onreadystatechange = function() {
  if (req.readyState == 4) {
    if (req.status == 200) {
      var ergebnis = req.responseText;
      document.getElementById("austausch").innerHTML = ergebnis;
    }
  }
};
req.open("GET", url, true);
req.send(null);
```

Wert	Bezeichnung	Bedeutung
0	UNINITIALIZED	Das Objekt wurde noch nicht initialisiert, d. h., es erfolgte noch kein Aufruf der Funktion `OPEN`.
1	LOADING	Das Request-Objekt wurde initialisiert, aber der Request noch nicht abgesetzt (mittels `SEND`).
2	LOADED	Der Request wurde mittels der Funktion `SEND` -abgesetzt.
3	INTERACTIVE	Teile der Antwort sind bereits verfügbar. Über das Feld `RESPONSETEXT` kann auf die empfangenen Daten zugegriffen werden.
4	COMPLETED	Die Bearbeitung des Requests ist beendet.

Quelle: http://www.teialehrbuch.de/Kostenlose-Kurse/AJAX/20824-XMLHttpRequest.html

ok: kein Problem auf Client oder Server

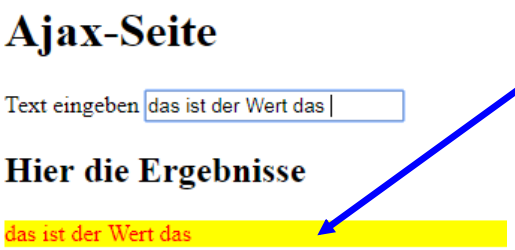
vom Server zurückgegebener Wert

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="ISO-8859-1">
        <title>Insert title here</title>
    </head>
    <script type="text/javascript">
        function ausgeben() {

            var url = "http://localhost:8080/SOTE2-2018-04-27/AjaxServlet?wert=" + escape(eingabe.value);
            var req = new XMLHttpRequest();

            req.onreadystatechange = function() {
                if (req.readyState == 4) {
                    if (req.status == 200) {
                        var ergebnis = req.responseText;
                        document.getElementById("austausch").innerHTML = ergebnis;
                    }
                }
            };
            req.open("GET", url, true);
            req.send(null);
        }

    </script>
<body>
    <h1>Ajax-Seite</h1>
    <form>
        <p>Text eingeben <input type="text" id="eingabe" onkeyup="ausgeben()"></p>
    </form>
    <h2>Hier die Ergebnisse</h2>
    <div id="austausch">Hier steht gleich der neue Text</div>

</body>
</html>
```

Wert im Textfeld (input)

Siehe vorherige Seite

einfügen des vom Server zurückgegebenen Werts (asynchron, ohne Nachladen der Seite)



Zugehöriger Servlet-Code

```
11
12 @WebServlet("/AjaxServlet")
13 public class AjaxServlet extends HttpServlet {
14     private static final long serialVersionUID = 1L;
15
16     public AjaxServlet() {
17         super();
18     }
19
20     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
21         String fromClient = request.getParameter("wert");
22         System.out.println("From Client " + fromClient);
23         PrintWriter writer = response.getWriter();
24         writer.write("<p style='color:red; background-color:yellow'>" + fromClient + "</p>");
25     }
26
27
28 }
```

Servlet schickt dem Client (Browser > XMLHttpRequest-Objekt) HTML-Code zurück. Besser wäre eine Rückgabe nur von Daten z.B. im XML oder JSON-Format.

Ajax mit JQuery

```
    </head>
<body>
    <h1>Mit JQuery und Ajax</h1>
    <div id="austausch">Hier komemen gleich die Daten rein</div>
    <div id="austausch2">Hier komemen gleich die Daten aus JSON rein</div>
    <button id="button1">Bitte Daten aus Text laden</button>
    <button id="button2">Bitte Daten aus JSON laden</button>
    <script type="text/javascript">

        //Hier Daten aus Text holen
        $("#button1").click(function(){
            $("#austausch").load("./daten/daten.txt");
        });

        //Hier JSON Daten holen
        $("#button2").click(function(){
            $.ajax(
                {
                    url: "./daten/studis.json",
                    dataType: "json",
                    success: function(json) {
                        var ergebnis = "<ul>";
                        $.each(json.Studis, function(){
                            ergebnis += "<li>" + this['vorname'] + "</li>";
                        });

                        ergebnis += "</ul>";
                        $("#austausch2").html(ergebnis);
                    }
                }

            );
        });
    </script>
```

kurze Variante

```
{
    "Hochschule" : "HTWG",
    'Studis' : [
        {
            "vorname" : "Alex"
        },
        {
            "vorname": "Leoni"
        }
```

Daten können entweder auf Webserver liegen (statisch) oder dynamisch mit Servlet erzeugt werden

Variante mit JSON und each-Function