

Web 2

Les 4: POST

OP HET EXAMEN...

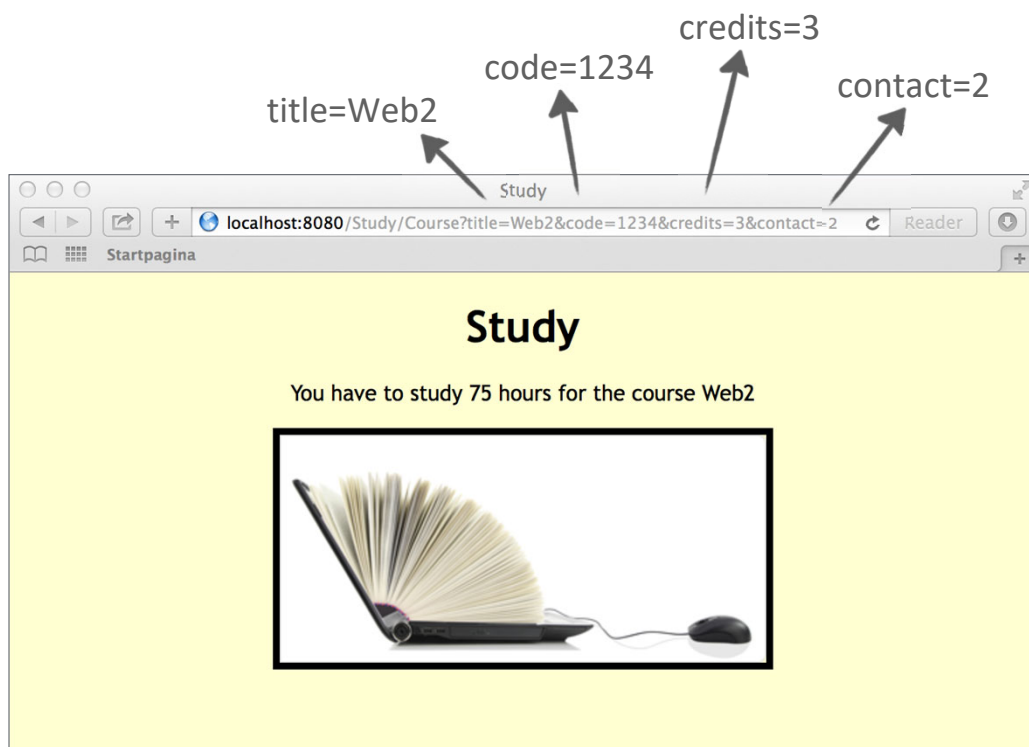


- ☑ Wat is het verschil tussen het attribuut *id* en het attribuut *name* bij een HTML input element?
- ☑ Waarvoor dient het attribuut *type* bij een HTML *input* element?
- ☑ Als ik een form submit, wordt er dan een HTTP GET of een POST uitgevoerd? Hoe komt dit?
- ☑ Waarheen wordt volgend formulier gestuurd:
`<form method="POST" action="#"> ?`
- ☑ Verklaar verschil tussen volgende lijnen:
 - `<form method="POST">`
 - `<form method="POST" action="Course">`
 - `<form method="POST" action="result.jsp">`
 - `<form method="GET" action="result.jsp">`
- ☑ Hoe worden parameters meegegeven in een POST request?
- ☑ Wanneer gebruik je een GET en wanneer een POST request?
- ☑ Noem 3 voor- en nadelen van een POST-request.
- ☑ Leg uit: een GET request moet idempotent zijn.
- ☑ Waarvoor wordt het bestand web.xml gebruikt.
- ☑ ...

AGENDA

- ❑ Herhaling
- ❑ POST
- ❑ web.xml





Ander voorbeeld: verwachte studielast berekenen

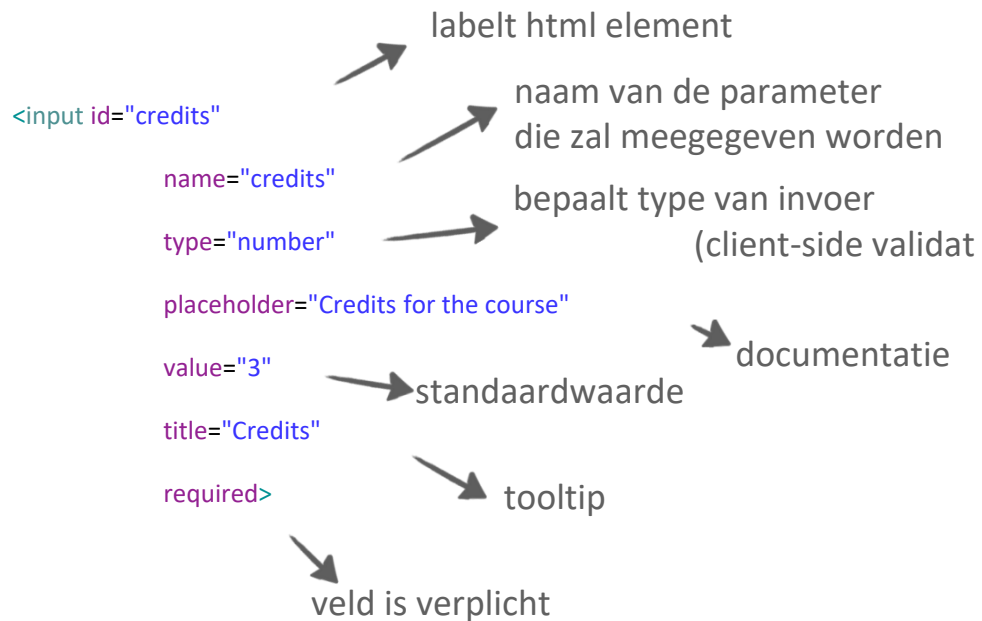
4 parameters —> URL werkt niet handig...

HTML FORM

WAAROM

- user input naar server sturen
- gebruiksvriendelijk
- browser bouwt URL op
- client-side validatie

INPUT ATTRIBUTEN



andere: size, maxlength, ...

zie

http://www.w3schools.com/tags/tag_input.asp

LABEL

- opschrift voor een input element
 - geen extra weergave functionaliteit
 - verhoogt usability en accessibility
- for-attribuut: verbindt opschrift met invoer

```
<label for="title">Name:</label>  
<input id="title" name="title">
```

klikken op label activeert control

label en input field kunnen al dan niet genest zijn:

- voordeel niet-genest: meer styling mogelijkheden
- voordeel genest:
 - for- en id- attribuut zijn overbodig
 - eenvoudige positionering van radiobutton of checkbox

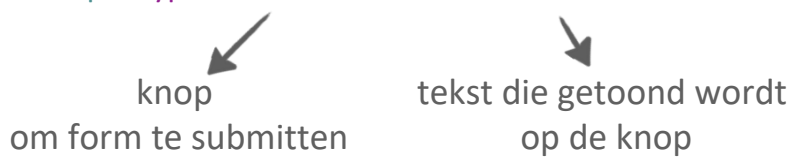
<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/label>

INPUT SUBMIT BUTTON

- `<input type="submit">`

- Voorbeeld:

`<input type="submit" value="Calculate">`



Waarom geen `name=""` attribuut?

- omdat dit geen parameter is die we willen meegeven met de request
- name attribuut kan toegevoegd worden en dan wordt er een parameter meegegeven met de value "Calculate"

`<button>` of `<input type="submit">`?

- button is veelzijdiger naar styling toe (geneste elementen)
- gedrag bij submit form is echter

browser-afhankelijk

- dus `<button>` eerder voor javascript en `<input>` voor form submit

AGENDA

- ☒ Herhaling
- ☐ POST
- ☐ web.xml



INPUT → SERVER



Welke HTTP request wordt verstuurd?

HTTP METHODE

- form-attribuut `method`
- `<form method="HTTP methode" >`
- voorbeeld:
`<form method="GET">`
...
- standaard: GET
- GET: parameters meegegeven in request-**URL**

```
http://localhost:8080/Study/Course?title=BOP&code=1234  
&credits=6&contact=4
```

```
...  
<body>  
<h2>Course info</h2>  
  <form method="GET" action="Course">  
    <fieldset>  
      <legend>Identification</legend>  
      <p>  
        <label for="title">Name: </label>  
        <input id="title" name="title">  
      </p>  
      <p>  
        <label for="code">Code (required): </label>  
        <input id="code" name="code" required>  
      </p>  
    </fieldset>  
  </form>  
</body>  
</html>
```

Course info

Identification

Name: BOP

Code (required): 1234

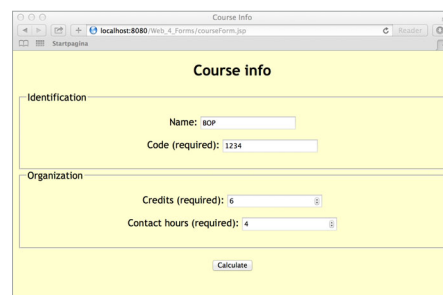
Organization

De waarde van het attribuut **name** van het input element wordt de **naam** van de parameter die meegegeven wordt met de querystring

Datgene wat de gebruiker invult in het formulier wordt als **waarde** van deze parameter meegegeven met de querystring

GET /Study/Course?title=BOP&code=1234&credits=6&contact=4 HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:31.0) Gecko/20100101 Firefox/31.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: nl-be,en-gb;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://localhost:8080/Study/courseForm.jsp
Cookie: JSESSIONID=08E34B1AF6FE4D802A0F3B9FC6F164C4; textwraapon=false; wysiwyg=textarea
Connection: keep-alive

HTTP GET REQUEST



Course info

Identification

Name: aop

Code (required): 1234

Organization

Credits (required): 6

Contact hours (required): 4

Calculate

- Indien GET als method wordt gekozen
- wordt er dus een GET request verstuurd en
 - zullen de parameters meegegeven worden als querystring in de **URL**
 - **LET OP:** als het attribuut "action" reeds een querystring bevat, wordt die overschreven met de parameters van het formulier (oplossing: hidden input fields <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/hidden>)

ALTERNATIEF: POST

- voorbeeld:

```
<form method="POST">  
...
```

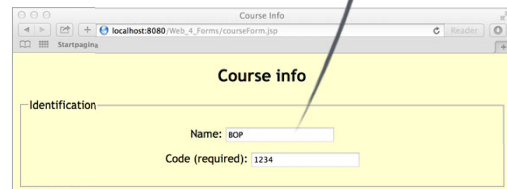
- parameters meegegeven in request-**body**

[Demo met Developer Tool]

http://localhost:8080/Study/Course

???

```
...  
<body>  
<h2>Course info</h2>  
  <form method="POST" action="Course">  
    <fieldset>  
      <legend>Identification</legend>  
      <p>  
        <label for="title">Name: </label>  
        <input id="title" name="title">  
      </p>  
      <p>  
        <label for="code">Code (required): </label>  
        <input id="code" name="code" required>  
      </p>  
    </fieldset>  
  </form>  
</body>  
</html>
```



De waarde van het attribuut **name** van het input element wordt de **naam** van de parameter die meegegeven wordt in de body van het bericht

Datgene wat de gebruiker invult in het formulier wordt als **waarde** van deze parameter meegegeven in de body van het bericht

POST /Study/Course HTTP/1.1

Host: localhost:8080

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:31.0) Gecko/20100101 Firefox/31.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: nl-be,en-gb;q=0.7,en;q=0.3

Accept-Encoding: gzip, deflate

Referer: http://localhost:8080/Study/courseForm.jsp

Cookie: JSESSIONID=08E34B1AF6FE4D802A0F3B9FC6F164C4; textwraapon=false; wysiwyg=textarea

Connection: keep-alive

title=BOP&code=1234&credits=6&contact=4

HTTP POST REQUEST

Course info

Identification

Name: aop

Code (required): 1234

Organization

Credits (required): 6

Contact hours (required): 4

Calculate

Indien POST als method wordt gekozen

- wordt er dus een POST request verstuurd en
- zullen de parameters meegegeven worden in de **body** van het bericht

```

@WebServlet("/Course")
public class CourseServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request,
                           HttpServletResponse response)
        throws ServletException, IOException {
        String title = request.getParameter("title");
        String code = request.getParameter("code");
        String creditsFromParameter = request.getParameter("credits");
        int credits = Integer.parseInt(creditsFromParameter);
        String contactFromParameter = request.getParameter("contact");
        int contactHours = Integer.parseInt(contactFromParameter);

        Course course = new Course(title, code, credits, contactHours);
        request.setAttribute("course", course);
        RequestDispatcher view =
            request.getRequestDispatcher("result.jsp");
        view.forward(request, response);
    }
}

```

CourseServlet.java

In de servlet overschrijf je deze keer de doPost methode i.p.v. de doGet() methode

	GET	POST
Bookmark	V	X
Cache	V	X
Restrictions on length	V	X
Restrictions on type	V	X
Back button	V	X (warning)
Parameters in history	V	X
Security	X	V
Visibility	V	X

V heeft en is goed
 V heeft maar is niet goed
 X heeft niet en is goed
 X heeft niet en is niet goed


 Useable


 Safe

Verschil tussen GET- en POST request:
<https://www.diffen.com/difference/GET-vs-POST-HTTP-Requests>

groene V	heeft en is goed
rode V	heeft maar is niet goed
groene X	heeft niet en is goed
rode X	heeft niet en is niet goed

CONVENTIE

IMPORTANT

- GET
 - GEEN side-effect
 - voor navigatie, ophalen gegevens, ...
 - één of 100 keer sturen maakt niet uit



IDEMPOTENT !

- POST
 - wel side-effect
 - voor toevoegen, wijzigen, verwijderen, ...

Dit onderscheid is een afspraak, wordt niet technisch afgedwongen.
Het is de verantwoordelijkheid van de ontwikkelaar om zich hieraan te houden.

NAVIGATIE

- form-attribuut `action`
- `<form action="URL">`
- Indien leeg → zelfde pagina
- Voorbeeld: `<form action="Course">`
...

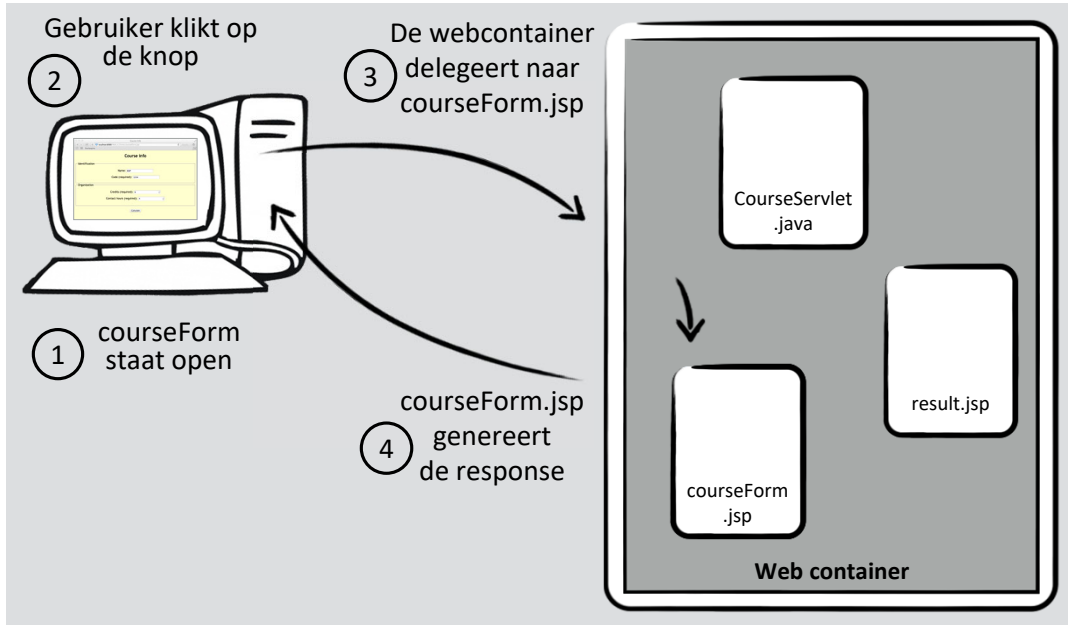
Path naar bv. servlet

`http://localhost:8080/Study/Course`

```
@WebServlet("/Course")
public class CourseServlet
private static final
protected void doPost
String title =
String code =
```

```
<form method="POST" action="#">
```

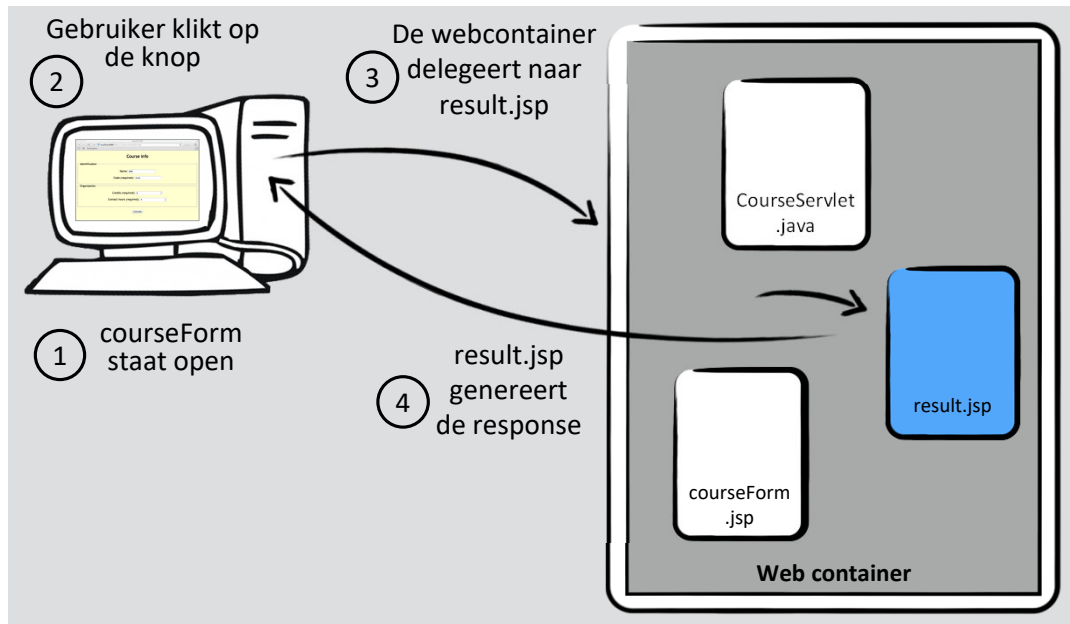
POST request naar dezelfde pagina



Op dit moment niet zo nuttig: alle logica om het resultaat te berekenen en te tonen zou dan bv. ook in courseForm.jsp moeten zitten —> courseForm zou niet erg herbruikbaar zijn en er zou te veel java in zitten.

```
<form method="POST" action="result.jsp">
```

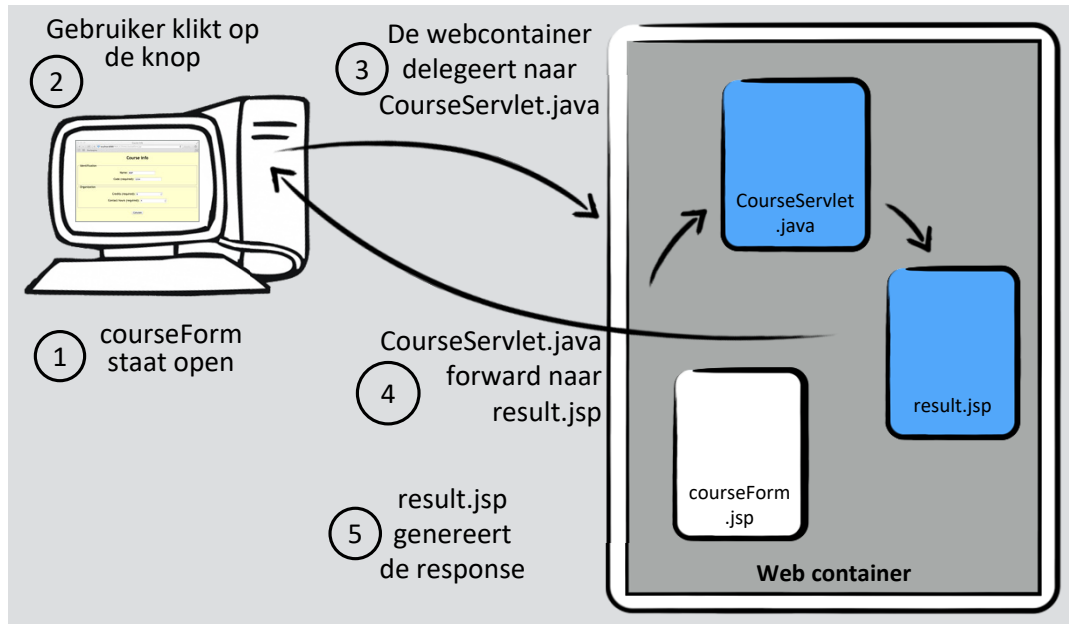
POST request naar jsp pagina



Alle logica om het resultaat te berekenen en te tonen zou dan bv. in result.jsp moeten zitten —> beter, want courseForm zou wel herbruikbaar zijn, maar we zouden te veel java in result.jsp hebben.

```
<form method="POST" action="Course">
```

POST request naar servlet met mapping 'Course'



Beste oplossing: beide jsp-pagina's zijn herbruikbaar en bevatten een minimum aan Java-code.

AGENDA

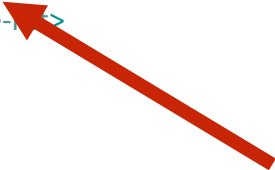
- ☒ Herhaling Web1
- ☒ POST
- ☐ web.xml



WEB.XML

- You use the web.xml file to define the start page of your web application.
- The web.xml file needs to be placed in the WEB-INF folder of your dynamic web project.
- then the web container uses this web.xml file to read properties defined in it and uses them to start your web application

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation=
"http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" version="2.5">
  <welcome-file-list>
    <welcome-file>courseForm.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```



Indicating which is the first
web page to land on when running
your web application

AGENDA

- ✓ Herhaling Web1
- ✓ POST
- ✓ web.xml



DEMO

[https://github.com/UCLLWeb2-2122/Demo MVC POST](https://github.com/UCLLWeb2-2122/Demo_MVC_POST)