

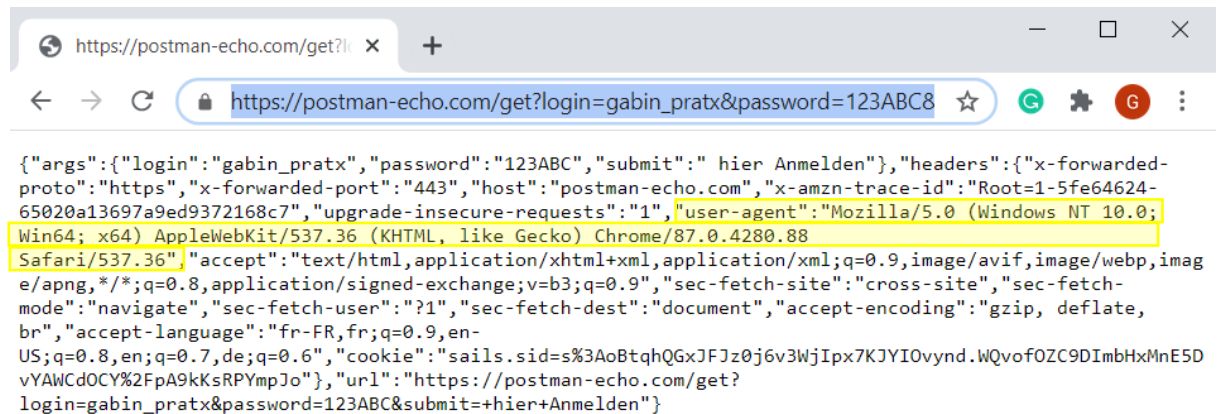
Markup & Html Praktikum

Fragen für die Abgabe

Aufgabe 1a

- Das HTML-Element `<form>` ist ein Feld eines Dokuments, der interaktive Steuerelemente enthält, mit denen ein Benutzer Daten an einen Webserver senden kann. Das `<method>` Attribut definiert die Methode (hier „GET“) mit der ein Server ein Request interpretieren soll. Die Methode der HTTP-GET-Methode ist dazu vorgesehen, Daten von einem bestimmter Quelle/Server abzurufen. Die Daten werden durch einen eindeutigen URI (Uniform Resource Identifier) identifiziert. Das `<action>`-Attribut enthält die URI der Webseite, an der die Daten übermittelt werden.
Umlaute und andere Sonderzeichen werden mit UTF-8 kodiert/dekodiert. UTF-8 überdeckt alle ASCII-Zeichen plus zusätzlich Nicht-ASCII-Zeichen wie Umlaute.
- Das HTML-Element `<fieldset>` wird verwendet, um mehrere interaktive Steuerelemente und Tags (`<label>`) in einem Webformular zu gruppieren. Der „Titel“ eines `<fieldset>` Elements wird mit dem Attribut `<legend>` definiert durch beispielsweise: `<legend>Login mit POST</legend>`.
- Wenn ein Label angeklickt wird erscheint im zugehörigen Textfeld ein Cursor und ein gelber Hintergrund.
- Die Label-Elemente und die Textfelder werden verknüpft indem das `<label>`-Element mit einem `<input>`-Element vom Typ „text“ verbunden werden. Dafür muss dem `<input>`-Element ein Identifier in Form eines `<id>`-Attributs übergeben werden. Der `<input>`-Element erhält einen `<for>`-Attribut.
- Das Attribut „name“ gibt den Namen des Datenfeldes an, das an die (über das Web-Formular) gesendeten Daten gebunden ist.
- Mit dem Attribut „value“ kann man Eingabewerte in einem Textfeld vordefinieren wie z.B.: `value=" hier Anmelden"`.

- Hier ein Beispiel eines gesendeten Formular mit der GET Methode:



```
{
  "args": {
    "login": "gabin_pratx",
    "password": "123ABC",
    "submit": " hier Anmelden"
  },
  "headers": {
    "x-forwarded-proto": "https",
    "x-forwarded-port": "443",
    "host": "postman-echo.com",
    "x-amzn-trace-id": "Root=1-5fe64624-65020a13697a9ed9372168c7",
    "upgrade-insecure-requests": "1",
    "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36",
    "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9",
    "sec-fetch-site": "cross-site",
    "sec-fetch-mode": "navigate",
    "sec-fetch-user": "?1",
    "sec-fetch-dest": "document",
    "accept-encoding": "gzip, deflate, br",
    "accept-language": "fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7,de;q=0.6",
    "cookie": "sails.sid=s%3AoBtqhQGxJFJz0j6v3WjIpx7KJYIOvynd.WQvofoZC9DImbHxMnESDvYAWCdOCY%2FpA9kKsRPYmpJo"
  },
  "url": "https://postman-echo.com/get?login=gabin_pratx&password=123ABC&submit=+hier+Anmelden"
}
```

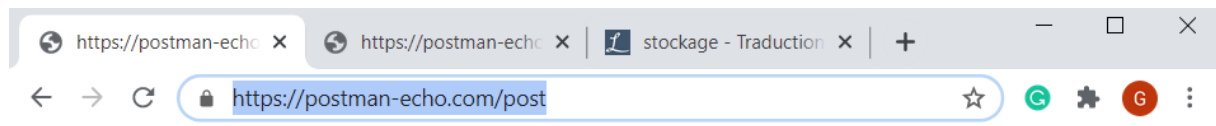
Man kann folgende Informationen herausfinden

- Die Anmeldedaten „args“ (login, password, submit)
- Daten über die Herkunft der Daten und das Datenhosting
- Den URL, die die Anmeldedaten beinhaltet

Die auf der oberen Abbildung in **Gelb** markierten Daten beinhalten Info über den Computer von dem die Daten gesendet wurden. Man findet unter anderem die Public IP Adresse meines Computers (87.0.4280.88).

- **User-agent:** Info über den Client, Betriebssystem, Browser, ...
- Host:** Info über den Server (Host)
- Content length:** Länge der Response zum Client
- Content type:** gibt den *media type* der Ressource an.
- Accept:** Beschreibt, welche Dateitypen erlaubt werden
- Scheme:** das https Protokoll
- Method:** hier GET, die http Methode um Daten an einem Server zu senden
- Status :** Das HTTP response code.

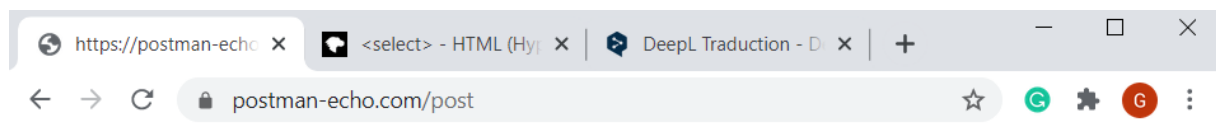
Aufgabe 1b



```
{
  "args": {},
  "data": "",
  "files": {},
  "form": {
    "login": "gabin_pratx",
    "password": "ABC123",
    "submit": "Anmelden"
  },
  "headers": {
    "x-forwarded-proto": "https",
    "x-forwarded-port": "443",
    "host": "postman-echo.com",
    "x-amzn-trace-id": "Root=1-5fe669fa-1388f72e0355c12a6fa80684",
    "content-length": "52",
    "cache-control": "max-age=0",
    "upgrade-insecure-requests": "1",
    "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36",
    "origin": "null",
    "content-type": "application/x-www-form-urlencoded",
    "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9",
    "sec-fetch-site": "cross-site",
    "sec-fetch-mode": "navigate",
    "sec-fetch-user": "?1",
    "sec-fetch-dest": "document",
    "accept-encoding": "gzip, deflate, br",
    "accept-language": "fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7,de;q=0.6",
    "if-none-match": "W/\\\"3eb-OsYs+d3Sohyhlv0C1kiD+cPo42M\\\"",
    "cookie": "sails.sid=s%3AWSblhTG5SNPUhHO5rCZ1NBZXPQm70TB.R1mwIij5o%2BNd86u hhEsHNchoN8KqaA2g532%2FGQsKXMc",
    "json": {}
  },
  "login": "gabin_pratx",
  "password": "ABC123",
  "submit": "Anmelden",
  "url": "https://postman-echo.com/post"
}
```

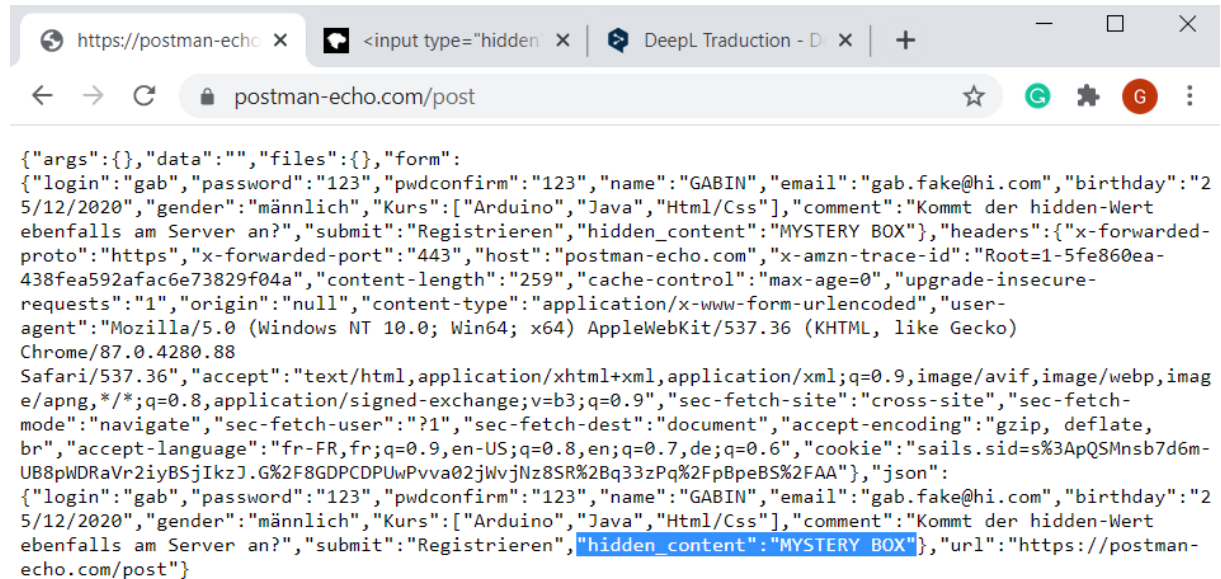
- Es gibt zwischen GET und POST folgende Unterschiede:
 - Die gesendeten Daten tauchen bei POST im Gegensatz zu GET in der URL nicht auf.
 -
- GET ist im Vergleich zu POST weniger sicher, da die gesendeten Daten Teil der URL sind. Für eine solche Anwendung wie hier (hier werden beim Anmelden Passwort und Benutzername gesendet) würde ich POST bevorzugen.

Aufgabe 1c



```
{
  "args": {},
  "data": "",
  "files": {},
  "form": {
    "login": "gabin",
    "password": "12AB34CD",
    "pwdconfirm": "12AB34CD",
    "name": "Weihnachtsmann",
    "email": "père.noël@ohohoh.fr",
    "birthday": "25/12/2020",
    "gender": "männlich",
    "Kurs": [
      "Arduino",
      "Java",
      "Matlab"
    ],
    "comment": "Frohe Festtage !",
    "submit": "Registrieren"
  },
  "headers": {
    "x-forwarded-proto": "https",
    "x-forwarded-port": "443",
    "host": "postman-echo.com",
    "x-amzn-trace-id": "Root=1-5fe67885-5de268ea76846e7e77391233",
    "content-length": "234",
    "cache-control": "max-age=0",
    "upgrade-insecure-requests": "1",
    "origin": "null",
    "content-type": "application/x-www-form-urlencoded",
    "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36",
    "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9",
    "sec-fetch-site": "cross-site",
    "sec-fetch-mode": "navigate",
    "sec-fetch-user": "?1",
    "sec-fetch-dest": "document",
    "accept-encoding": "gzip, deflate, br",
    "accept-language": "fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7,de;q=0.6",
    "json": {}
  },
  "login": "gabin",
  "password": "12AB34CD",
  "pwdconfirm": "12AB34CD",
  "name": "Weihnachtsmann",
  "email": "père.noël@ohohoh.fr",
  "birthday": "25/12/2020",
  "gender": "männlich",
  "Kurs": [
    "Arduino",
    "Java",
    "Matlab"
  ],
  "comment": "Frohe Festtage !",
  "submit": "Registrieren",
  "url": "https://postman-echo.com/post"
}
```

- Mit dem Attribut „multiple“, das auf HIGH (Boolean) gesetzt wird, kann man mehrere Elemente selektieren. Die selektierten Werte werden dann in einem Array zusammengefasst und zwischen „[]“ ausgegeben, hier ein Beispiel:
"Kurs":["Arduino","Java","Matlab"]



```
{
  "args": {},
  "data": "",
  "files": {},
  "form": {
    "login": "gab",
    "password": "123",
    "pwdconfirm": "123",
    "name": "GABIN",
    "email": "gab.fake@hi.com",
    "birthday": "25/12/2020",
    "gender": "m\u00e4nnlich",
    "Kurs": ["Arduino", "Java", "Html/Css"],
    "comment": "Kommt der hidden-Wert ebenfalls am Server an?",
    "submit": "Registrieren",
    "hidden_content": "MYSTERY BOX",
    "headers": {
      "x-forwarded-proto": "https",
      "x-forwarded-port": "443",
      "host": "postman-echo.com",
      "x-amzn-trace-id": "Root=1-5fe860ea-438fea592afac6e73829f04a",
      "content-length": "259",
      "cache-control": "max-age=0",
      "upgrade-insecure-requests": "1",
      "origin": "null",
      "content-type": "application/x-www-form-urlencoded",
      "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36",
      "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9",
      "sec-fetch-site": "cross-site",
      "sec-fetch-mode": "navigate",
      "sec-fetch-user": "?1",
      "sec-fetch-dest": "document",
      "accept-encoding": "gzip, deflate, br",
      "accept-language": "fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7,de;q=0.6",
      "cookie": "sails.sid=s%3ApoQSMnsb7d6m-UB8pwDRaVr2iyBSjIkzJ.G%2F8GDPCDPuWpVva02jWvjNz8SR%2Bq33zPq%2FpBpeBS%2FAA",
      "json": {
        "login": "gab",
        "password": "123",
        "pwdconfirm": "123",
        "name": "GABIN",
        "email": "gab.fake@hi.com",
        "birthday": "25/12/2020",
        "gender": "m\u00e4nnlich",
        "Kurs": ["Arduino", "Java", "Html/Css"],
        "comment": "Kommt der hidden-Wert ebenfalls am Server an?",
        "submit": "Registrieren",
        "hidden_content": "MYSTERY BOX",
        "url": "https://postman-echo.com/post"
      }
    }
  }
}
```

Der Feld vom Typ “hidden” taucht im Registrierungsformular nicht auf. Es kommt jedoch an dem Server an !

Mit einem hidden Feld k\u00f6nnen Webentwickler Daten einbinden, die von Benutzern nicht gesehen oder ge\u00e4ndert werden k\u00f6nnen, wenn ein Formular abgeschickt wird.

In einem hidden Feld wird oft gespeichert, welcher Datensatz in der Datenbank aktualisiert werden muss, wenn das Formular abgeschickt wird. Des Weiteren kann der Wert f\u00fcr den Benutzer nicht im Inhalt der Seite angezeigt werden, er ist jedoch mit dem Entwicklertool des Browsers oder der Funktion "Quelltext anzeigen" sichtbar (und kann bearbeitet werden).

Aufgabe 1d

- Die CSS-Pseudo-Klasse **:valid** zeigt jedes `<input>`-Element an, dessen Inhalt erfolgreich validiert wurde. Dadurch kann der Benutzer sehen, wenn ein Feld einen bestimmten Layout annimmt (hier gr\u00fcn wenn richtig), dass seine Daten richtig eingegeben wurden.
- Mit dem Boolean Attribut **required** kann man verpflichten, dass der Benutzer einen Wert in einem bestimmten Feld eingibt, damit das entsprechende Formular gesendet werden kann.

Mit der CSS-Pseudo-Klasse **:required** kann man nun alle `<input>`-Element ansprechen, f\u00fcr die das Attribut **required** aktiviert ist. Damit kann man diese Elemente entsprechend formatieren mit z.B. einen roten Rahmen.

-

GET vs POST
Registrierung
HTML5

Registrierung in HTML5:

Logindaten

Benutzername:

Passwort:

Bestätigen: Veuillez renseigner ce champ.

Benutzerdaten

Name:

E-Mail:

Telefon:

Geburtsdag:

Alter:

Anz. Teilnehmer:

Geschlecht: ☐ Männlich ☐ Weiblich

Kurse:

Kommentar:

Abb. 1a

GET vs POST
Registrierung
HTML5

Registrierung in HTML5:

Logindaten

Benutzername:

Passwort:

Bestätigen:

Benutzerdaten

Vorname:

Nachname:

E-Mail: Veuillez respecter le format requis.

Telefon:

Geburtsdag:

Alter:

Anz. Teilnehmer:

Geschlecht: ☒ Männlich ☐ Weiblich

Kurse:

Kommentar:

Abb. 1b

Das Formular wird nicht geschickt solange alle obligatorische Felder nicht alle richtig ausgefüllt sind – siehe Abb. 1a – und solange alle vorformatierte Felder nicht validiert (mit dem richtigen Format ausgefüllt) sind – siehe Abb. 1b. Abbildung sieht.

- Mit dem Attribut **pattern** kann man für ein bestimmtes input-Feld eine bestimmte Formatierung verlangen. Mit dem Attribut **placeholder** kann man zusätzlich dieses input-Feld mit einem Musterinhalt belegen. Hier ein Beispiel:

```
type="text" id="nachname" name="surname" required pattern="[A-Z]{3,15}"
placeholder="NACHNAME"/>
```

Hier wird verlangt, dass der Nachname in 3 bis max. 15 Großbuchstaben angegeben wird wie z.B. „NACHNAME“.

- Das **autofocus** Attribut bestimmt, auf welches <input>-Element beim Laden der Seite automatisch fokussiert wird.

Aufgabe 2

In der SVG-Datei resistor.svg besteht das Symbol des variablen Widerstands aus 4 svg-Elemente: eine gerade Linie, ein Rechteck, eine schräge Linie und ein Polygon (Dreieck der Pfeilspitze).

Aufgabe 3

- Nein, damit das Design vom Header und von der Navigation unterschiedlich kontrolliert werden können.
- - **Ja** eventuell für die „*grüne box*“
 - **Nein** weil in einem Blog ändert sich ständig den Inhalt. Daher sollte ein Blog besser als *article* weiter organisiert werden. Der *main* Element beinhaltet i.d.R. den ganzen Inhalt einer Seite.