# Emailadressen als formale Sprache

Bei vielen Internetdiensten muss man sich mit einer *gültigen* Mailadresse anmelden. Aber wer entscheidet eigentlich, was eine gültige Mailadresse ist?

| | |
|---|---|
| Ungültig! | Gültig? |

Das Anmeldeformular besitzt bereits eine Komponente zur Validierung von E-Mail-Adressen. Diese Komponente prüft, ob die eingegebene Zeichenkette überhaupt eine Mailadresse sein kann, manchmal auch, ob die E-Mail-Adresse tatsächlich vergeben ist.

**(A1)**

**(a)** Führe selbst Experimente mit dem RFC822 email address validator aus. Versuche auf diese Weise herauszufinden, wie E-Mail-Adressen (nicht) aufgebaut werden dürfen.

**(b)** E-Mail-Adressen werden nach der RFC 822 überprüft. Recherchiere, was es mit der RFC 822 auf sich hat.

## Das eMail Format in RFC 822

Requests for Comments (kurz RFC) enthalten Festlegungen zur technischen und organisatorischen Konzeption des Internets.

So enthält die RFC 822 aus dem Jahr 1982 unter anderem sämtliche Festlegungen zum E-Mail-Format. In dieser RFC wird u.a. genau beschrieben, wie eine E-Mail-Adresse aufgebaut werden darf. Hier ein Auszug aus der RFC 822:

```
6. ADDRESS SPECIFICATION
6.1. SYNTAX

     address     =  mailbox                      ; one addressee
                 /  group                        ; named list
     group       =  phrase ":" [#mailbox] ";"
     mailbox     =  addr-spec                    ; simple address
                 /  phrase route-addr            ; name & addr-spec
     route-addr  =  "<" [route] addr-spec ">"
     route       =  1#("@" domain) ":"           ; path-relative
     addr-spec   =  local-part "@" domain        ; global address
     local-part  =  word *("." word)             ; uninterpreted
                                                 ; case-preserved
     domain      =  sub-domain *("." sub-domain)
     sub-domain  =  domain-ref / domain-literal
     domain-ref  =  atom                         ; symbolic reference
```

Du wirst nicht alle Details dieses RFC-Auszugs verstehen. Versuche dennoch, Teile dieser Adressbeschreibung zu deuten.

Um zu verstehen, wie solche Spezifikationen zu deuten sind, werden wir in den folgenden Abschnitten selbst eine Adress-Spezifikation erstellen - allerdings nur für stark vereinfachte E-Mail-Adressen.

# Einfache (eigene) E-Mail-Adressen

Es gibt eine Vielzahl an Möglichkeiten, wie E-Mail-Adressen aufgebaut sein können. Beispiele - auch ungewöhnliche - findest du auf den Seiten von Wikipedia.

Wir werden uns hier nicht mit all diesen Adressformaten beschäftigen. Ziel dieses Abschnittes ist es, ein Verfahren zur präzisen Festlegung des Aufbaus von E-Mail-Adressen einzuführen. Für diesen Zweck reicht es, ein sehr einfaches E-Mail-Adressformat zu betrachten.

## Unsere Einfach-Mailadressen

Wir betrachten nur vereinfachte E-Mail-Adressen, in denen nur die Symbole `b`, `@` und `.` vorkommen dürfen.

Beispiel: `bb@bbb.bb`

Folgende Regeln sollen zur Bildung solcher E-Mail-Adressen beachtet werden:

- Eine vereinfachte E-Mail-Adresse besteht aus einem User-Namen gefolgt vom @-Symbol und einer Domain-Angabe.
- Der User-Name soll nur aus b´s bestehen.
- Die Domainangabe soll aus Subdomains und einer Topleveldomain aufgebaut sein, die jeweils mit einem Punkt getrennt werden.
- Eine Subdomain und eine Topleveldomain besteht nur aus b´s.

### (A2)

Welche der folgenden Zeichenketten stellen vereinfachte E-Mail-Adressen dar?

```
b@b.bbb
@b.b.bb
bbb@bbbb
bb.b@b.bb
```

Warum ist es in einigen Fällen schwierig, das zu entscheiden?

### Eine formale Beschreibung mit Syntaxdiagrammen

Informelle Beschreibungen sind oft nicht so präzise, dass keine Zweifelsfälle entstehen können. In der

Informatik ist es daher üblich, Festlegungen formal zu beschreiben.

Die folgenden Syntaxdiagramme sollen den Aufbau unserer vereinfachten E-Mail-Adressen präzise festlegen.

**Emailadresse:**

**User:**

**Domain:**

**Subdomains:**

**Topleveldomain:**

**Name:**

**Buchstabe:**

**(A3)**

Entscheide mit Hilfe der Syntaxdiagramme, welche der folgenden Zeichenketten vereinfachte E-Mail-Adressen darstellen und welche nicht.

```
b@b.bbb
@b.b.bb
bbb@bbbb
bb.b@b.bb
```

# Vom Syntaxdiagramm zu den Regeln einer Grammatik

Gesucht ist eine Grammatik G=(V,Σ,P,S), die unsere Mailadressen als formale Sprache beschreibt.

**(A4)** Gib das Alphabet Σ unserer Sprache an.

## Herleitung

Nun benötigt man Regeln und Variablen. Dazu übersetzt man die Darstellung im Syntaxdiagramm in eine vereinfachte Zuordnung.

| | **Syntax** | **Regel** |
|---|---|---|
| Emailadresse: | | (1) S → U@D |
| User: | | (2) U → N |
| Domain: | | (3) D → KT |
| Subdomains: | | (4) K → N.<br>(5) K → N.K |
| Topleveldomain: | | (6) K → N |
| Name: | | (7) N → B<br>(8) N → BN |
| Buchstabe: | | (9) B → b |

Man sieht, dass „Schleifen“ im Syntaxdiagramm zu Rekursionen im Regelwerk werden. Die doppelten Regeln bedeuten jeweils „entweder Regel … oder Regel …“, man kann verkürzt auch schreiben: `(4) K → N . | N . K`. Nun haben wir die Menge der Variablen (`S, U, D, K, T, N, B`), die Regeln („**P**roduktionen“, in der Tabelle rechts) und die Startregel (S), unsere Grammatik ist also komplett.

## Anwendung: Ableiten von Worten anhand der Grammatik

Die Ableitung der E-Mail-Adresse `bb@b.bbb.bb` kann man nun wie folgt mit Ersetzungsvorgängen beschreiben[1)]:

```
S ->                                    # (1)
U @ D ->                                # (2)
N @ D ->                                # (8)
B N @ D ->                              # (9)
b N @ D ->                              # (7)
b B @ D ->                              # (9)
b b @ D ->                              # (3)
b b @ K T ->                            # (5)
b b @ N . K T ->                        # (7)
b b @ B . K T ->                        # (9)
b b @ b . K T ->                        # (4)
b b @ b . N . T ->                      # (8)
b b @ b . B N . T ->                    # (9)
b b @ b . b N . T ->                    # (8)
b b @ b . b B N . T ->                  # (9)
b b @ b . b b N . T ->                  # (7)
b b @ b . b b B . T ->                  # (9)
b b @ b . b b b . T ->                  # (6)
b b @ b . b b b . N ->                  # (8)
b b @ b . b b b . B N ->                # (9)
b b @ b . b b b . b N ->                # (7)
b b @ b . b b b . b B ->                # (9)
```

```
b b @ b . b b b . b b
```

**(A5)**

- Leite die Mailadresse b@bb.b anhand unserer Grammatik ab.
- Mache dir klar, dass man b.b@bbb.bund b@bb nicht ableiten kann, weshalb das kein gültigen Worte unserer Sprache sind.

## Experimente mit JFlap

Mit Hilfe von JFlap (Download) kann man auch Grammatiken experimentell testen. Am Beispiel unserer Grammatik für vereinfachte E-Mail-Adressen soll dies hier gezeigt werden.

**(A5)**

Starte JFlap, wähle den Menupunkt `[Grammar]` aus und gib die Regeln unserer Grammatik im Grammatik-Editor ein.Variablen dürfen nur aus einem Buchstaben bestehen, Leerzeichen werden von JFLAP beachtet - wo keine sein sollen, dürfen auch keine sein.

---

Dieser Abschnitt ist auf Basis der Seite https://www.inf-schule.de/sprachen/sprachenundautomaten/sprachbeschreibung/grammatiken/fallstudie_email in inf-schule.de entstanden. Lizenz: CC-BY-SA

1)
Leerzeichen zur übersichtlicheren Darstellung

From:
https://wiki.qg-moessingen.de/ - **QG Wiki**

Permanent link:
**https://wiki.qg-moessingen.de/faecher:informatik:oberstufe:automaten:formale_sprachen:mailadressen:start?rev=1601990367**

Last update: **06.10.2020 15:19**