Dokumentation des PoCs zur Mailverifizierung

TLDs crawlen

Um eine List von allen TLDs der Hochschulen zu bekommen, wurde eine stetig gepflegte Liste aller Hochschulen[1] auf Wikipedia gefunden. Dort verlinkt jeder Eintrag einer Hochschule auf eine Wiki-Seite zur Hochschule. Diese Wiki-Seiten sind gleichmäßig aufgebaut: sie beinhalten (fast) immer eine Infobox, in der ein externer Link steht. Dieser externe Link ist die Verlinkung zur Hochschule.

Somit wurde ein Crawler in Node.js programmiert, der:

1. alle URLs zu den Wikipedia-Seiten der Hochschule findet
2. aus jeder Wikipedia-Seite den Link zur Hochschule extrahiert

Voraussetzungen der Seiten:

- Die Hochschulliste hat nur eine `<table>` mit einer `<tr>` pro Hochschuleintrag
- Im ersten `<td>` des Hochschuleintrags der Liste steht die Verlinkung
- Die Hochschul-Wiki-Seite hat ein Element mit dem Selektor `#Vorlage_Infobox_Hochschule`
- In diesem Element ist ein Link mit dem Selektor `.external`

Ablauf ist

1. Hochschulliste aufrufen
2. Alle Links zu den Hochschul-Wikipedia-Seiten extrahieren
3. Alle Hochschul-Wikipedia-Seiten nacheinander aufrufen
4. Die Infobox extrahieren
5. Aus der Infobox den Link extrahieren
6. Die Verlinkung zu einer TLD normalisieren (ohne Pfad und Protokoll)
7. Aus allen TLDs ein Array generieren
8. Array als JSON in Datei speichern

Dieses Verhalten ist nicht 100% fail-safe und es können nicht alle 425 Hochschul-TLDs gefunden werden. Jedoch werden in unseren Tests 422 TLDs gefunden und nur 3 nicht gefunden.

Die Ausgabe sieht wiefolgt aus:

```
[
    "nrw.de",
    "diploma.de",
    "dekra-hochschule-berlin.de",
```

[1] https://de.wikipedia.org/wiki/Liste_der_Hochschulen_in_Deutschland

```
  "hs-aalen.de",
  "fhvd.de",
  "businessschool-berlin-potsdam.de",
  "design-akademie-berlin.de",
  "bbw-hochschule.de",
  "bsb-hochschule.de",
...
```

Dabei wurde festgestellt, dass auch die Subdomain zur TLD hinzugefügt werden muss, um teilweise spezifischere Domains als *nrw.de* zu erreichen.

Diese Datei kann dann vom Client abgerufen werden.

User Input Validieren

Hier war der erste Schritt, den Input des Benutzers aus dem EditText Feld zu extrahieren.

```
final EditText editText = (EditText) findViewById(R.id.email_address);
editText.setOnEditorActionListener(new TextView.OnEditorActionListener() {
    @Override
    public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
        boolean handled = false;
        // No submit button, check if submit on keyboard is pressed
        if (actionId == EditorInfo.IME_ACTION_SEND) {
            userMail = (String) editText.getText().toString();

            // Debug
            System.out.println(editText.getText().toString());

            getJSON();
            handled = true;
        }
        return handled;
    }
});
```

Im nächsten Schritt wird eine Funktion aufruft, die einen GET-Request auf die TLD-Ressource ausführt, die vom oben beschriebenen Crawler bereit gestellt wird (für unsere Zwecke ist diese noch über Github gehostet).

```
public void getJSON () {
    // Instantiate the RequestQueue.
    RequestQueue queue = Volley.newRequestQueue(this);
    String url ="https://raw.githubusercontent.com/Plsr/EISWS1516MaemeckePoplawski/master/MS3/PoC/Source/PoC3/assets/tlds

    // Request a string response from the provided URL.
    StringRequest stringRequest = new StringRequest(Request.Method.GET, url,
            new Response.Listener<String>() {
                @Override
                public void onResponse(String response) {
                    try {
                        jsonArray = new JSONArray(response);
                        setResult();


                    } catch (JSONException e) {
                        e.printStackTrace();
                    }

                }
            }, (error) -> { System.out.println("That didn't work!"); });
    // Add the request to the RequestQueue.
    queue.add(stringRequest);
}
```

Ist der GET-Request ein Erfolg, wird die response zur weiteren Verarbeitung in ein JSONArray umgeformt und in der Methode setResult() zur Validierung der eingegebenen Mailadresse verwendet.
Die Methode setResult() fragt lediglich über die Methode checKTLD() ab, ob die eingegebene Adresse valide ist und setzt den entsprechenden Indikator.

In der Funktion checkTLD() wird mit Hilfe einer Regular Expression überprüft, ob die TLD in der eingegebenen Mailadresse mit einer der TLDs übereinstimmt, die der Crawler liefert oder nicht und gibt den entsprechenden boolschen Wert zurück.