

Sukunimi **Etunimi** **Merkinnät**

Rinta-Paavola Juho **Aalto** **Edistysaskeleita** **opiskelijoiden** **tagaamisessa** Lisää merkintä ▾

Mitä?

- A+

- +5933 -759 =5174 riviä koodia
- Tagien kokonaisvaltainen uudistus
- Mahdollisuus kansainvälistää JavaScriptiä
- Kasa pienempiä ominaisuuksia ja bugfixejä

Mitä?

- Pieniä parannuksia a-plus-client, rst-tools, django-colortag, jsvee, mooc-grader, O1 oppimateriaali
- Skripti, jolla voi lisätä tageja automaattisesti
- Neuvontajonon englanninnos

Mitä tageilla pystyy nykyään tekemään?

Mitä tageilla pystyi ennen tekemään?

Mitä tageilla pystyy nykyään tekemään?

Tagien tekstiä voi lukea!

Filter users: Aalto MOOC Bacon Eggs

Filter users: Aalto MOOC Bacon Eggs Spam

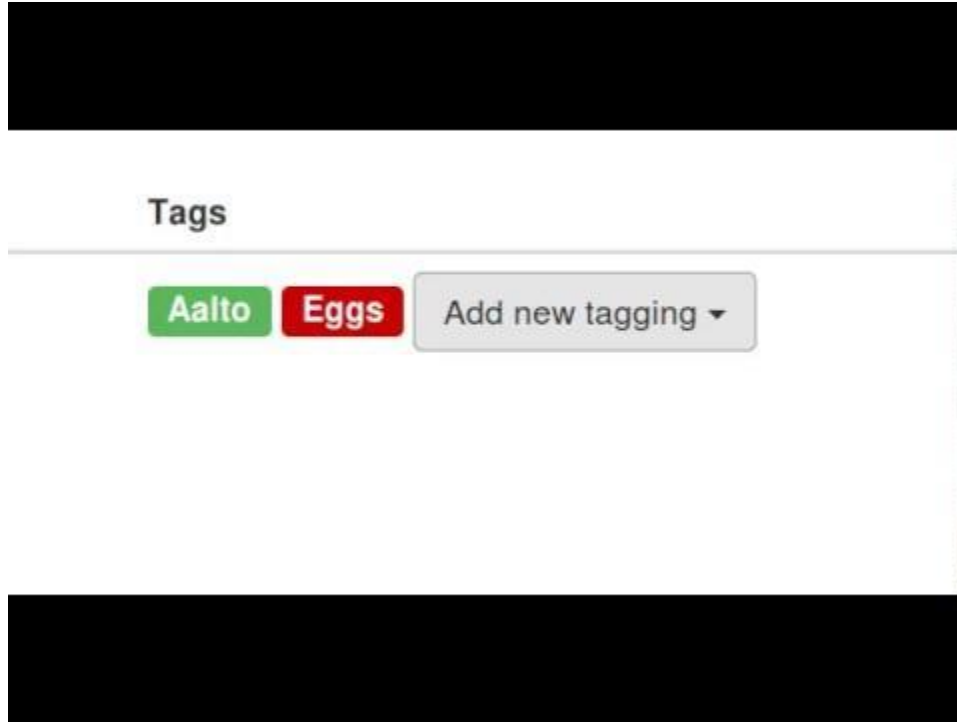
Tags

Aalto Bacon Eggs Spam

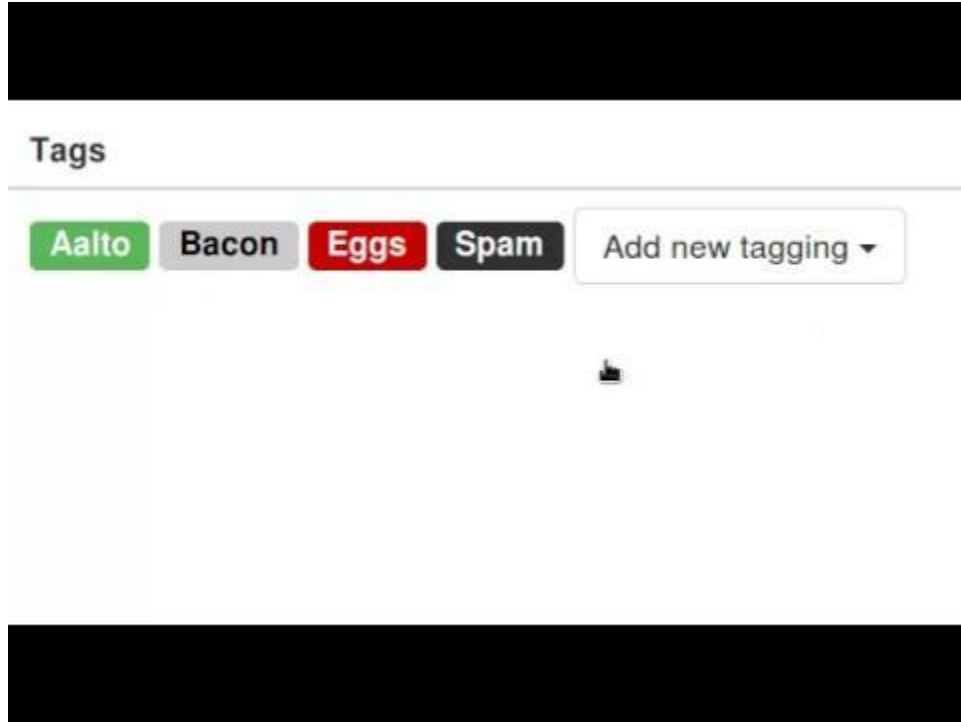
Tags

Aalto Bacon Eggs Spam

Tageja voi lisätä!



Tageja voi poistaa!

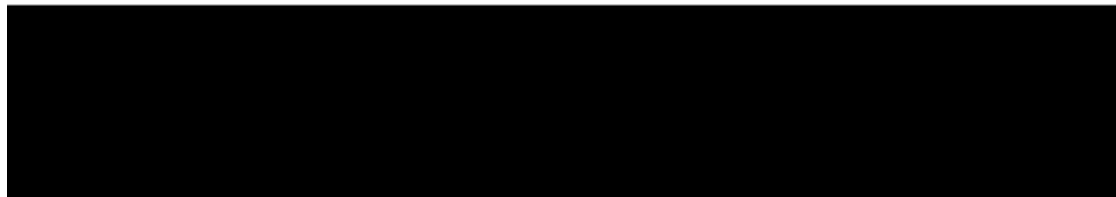


Voi lisätä ja poistaa monelle opiskelijalle kerralla!



Number of students 3 of which selected 0

<input type="checkbox"/> Student id ▼	Last name	First name	Email	Tags
<input type="checkbox"/>	Robinson	Ruth	root@localhost	Aalto Bacon Spam Add new tagging ▾
<input type="checkbox"/>	Bar	Foo	foo1	Aalto Spam Add new tagging ▾
<input type="checkbox"/>	Gruitt	Qux	foo2	Aalto Spam Add new tagging ▾



Kurssimateriaalia voi kustomoida tagien perusteella!

o1 / 3. Week 3 / 3.5 Chapter 3.4: Eggs, Bacon, and Spam

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Vaihda suomeksi

Chapter 3.4: Eggs, Bacon, and Spam

suomenkielinen

o1 / 3. Kierros 3 / 3.5 Luku 3.4: Kananmunia, pekonia sekä sian- ja naudanlihasäilykettä

Kananmunia, pekonia sekä sian- ja naudanlihasäilykettä

Tageja voi lisätä tehtävien vastausten perusteella!

```
tag_slugs = (tag_for_form_value[field[0]][field[1]]
              for field in submission_data
              if field[0] in tag_for_form_value)
user_ids = (submitter['id'] for submitter in submitters)
post_dataset = (
    {
        'user': {
            'id': user_id,
        },
        'tag': {
            'slug': tag_slug,
        },
    }
    for user_id in user_ids
    for tag_slug in tag_slugs)
for data in post_dataset:
    api.do_post('{courses_url}{course_id}[taggings_url]'
               .format(**CONFIG), json=data)
```

```
class APlusCourseHookHTTPRequestHandler(BaseHTTPRequestHandler):
    def do_POST(self):
        content_length = int(self.headers['Content-Length'])
        post_data = parse_qs(self.rfile.read(content_length).decode('utf-8'))
        exercise_id, *_ = (int(id) for id in post_data['exercise_id'])
        submission_id, *_ = (int(id) for id in post_data['submission_id'])
        # Wait before making requests to A+, because the submission is not ready
        # to be read from the API when A+ calls the hook
        Timer(CONFIG['wait_after_post'],
              add_tagging,
              (exercise_id, submission_id)).start()
        self.send_response(204)
```

```
INFO:requests.packages.urllib3.connectionpool:Resetting dropped con
nection: localhost
DEBUG:requests.packages.urllib3.connectionpool:"GET /api/v2/submiss
ions/108/ HTTP/1.1" 200 None
DEBUG:aplustag_client.client:making POST 'http://localhost:8000/api/v2
/courses/3/taggings/', headers={'Authorization': 'Token 14e94a47ccd
edf679728dadc9e388e35c32d00c', 'Accept': 'application/vnd.aplustag
on'}, params={}, data=None, json={'user': {'id': 1}, 'tag': {'slug':
'en'}}
INFO:requests.packages.urllib3.connectionpool:"POST /api/v2/course
s/3/taggings/ HTTP/1.1" 201 None
DEBUG:requests.packages.urllib3.connectionpool:Resetting dropped con
nection: localhost
INFO:Traceback (most recent call last):
  File "/usr/lib/python3.5/socketserver.py", line 232, in serve_for
ever
    ready = selector.select(poll_interval)
  File "/usr/lib/python3.5/selectors.py", line 376, in select
KeyboardInterrupt
```