API

- När vi gör ett anrop till en webbsida så innehåller svaret oftast HTML.
- HTML är bra för att strukturera upp vad information är för en browser, men det är inte ett bra sätt att hantera informationen programmeringsmässigt, det är svårt att få ut den informationen vi behöver.
- Kan man få andra typer av resultat?
  - https://randomuser.me/api/
- Application Programming Interface
- Låter oss hämta det data vi behöver i ett format som vi kan bearbeta, vanligen JSON eller XML.

JSON

- JSON (JavaScript Object Notation), är ett kompakt, textbaserat format som används för att utbyta data.

- Datatypen nedan är string, men påminner detta format om något?

```
{
  "firstName": "Jason",
  "lastName": "Smith",
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021"
  },
  "phoneNumber": [
    { "type": "home", "number": "212 555-1234" },
    { "type": "fax", "number": "646 555-4567" }
  ],
  "newSubscription": false,
  "companyName": null
}
```

- https://jsfiddle.net/emmio_micke/ahq6pzv9/

Fetch

- Det (just nu) nyaste sättet att hämta data från ett API är att använda `fetch`.

- `Fetch` körs asynkront och returnerar ett `Promise`.

- http://jsfiddle.net/emmio_micke/jhf2borz/
- http://jsfiddle.net/emmio_micke/8do61zy4/
- https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch

Uppgift

- Forka denna fiddle (eller kopiera bara koden) och ändra så att ni kan lägga till data från API:et istället för från formuläret.

- Om ni vill ha en extra utmaning, integrera lösningen med uppgiften vi gjorde i förra veckan där vi la till uppgifterna i en tabell och gav användaren möjlighet att radera rader och uppdatera uppgifter.

- http://jsfiddle.net/emmio_micke/h5j6dzbv/

API

- Vad finns det för API:er?
    - Massor av olika. Man kan hitta produktdatabaser, bussavgångar, betalningslösningar, information om kalenderhändelser, statistik, Chuck Norris-skämt…
    - Vissa är öppna för allmänheten, vissa används för internt bruk på företag, vissa kostar…
        - http://apikatalogen.se/
        - http://mashup.se/apikatalog/

Behörighet

- Ibland kan man behöva autentisera sig för att få tillgång till API:et.

- *Autentisering* (authentication) handlar om att bevisa att man är den man är (t ex en inloggningsfunktion).

- *Auktorisering* (authorization) handlar om huruvida användaren har rätt att göra något.

- Även om jag kan bevisa att jag är Micke, så är det inte säkert att Micke har rätt att ändra kunddata, t ex.

Autentisera

- Skydda data
- Skydda servern från överbelastning
- Statistik

Basic authentication

- Metod för att låta en HTTP-användare skicka namn och lösenord i ett request.

- Klienten autentiserar sig genom en header
  - `Authorization: Basic <credentials>`
  - Credentials är en base64-kodad sträng bestående av användare:lösenord.

- https://jsfiddle.net/emmio_micke/w9ftu2d6/

API-key

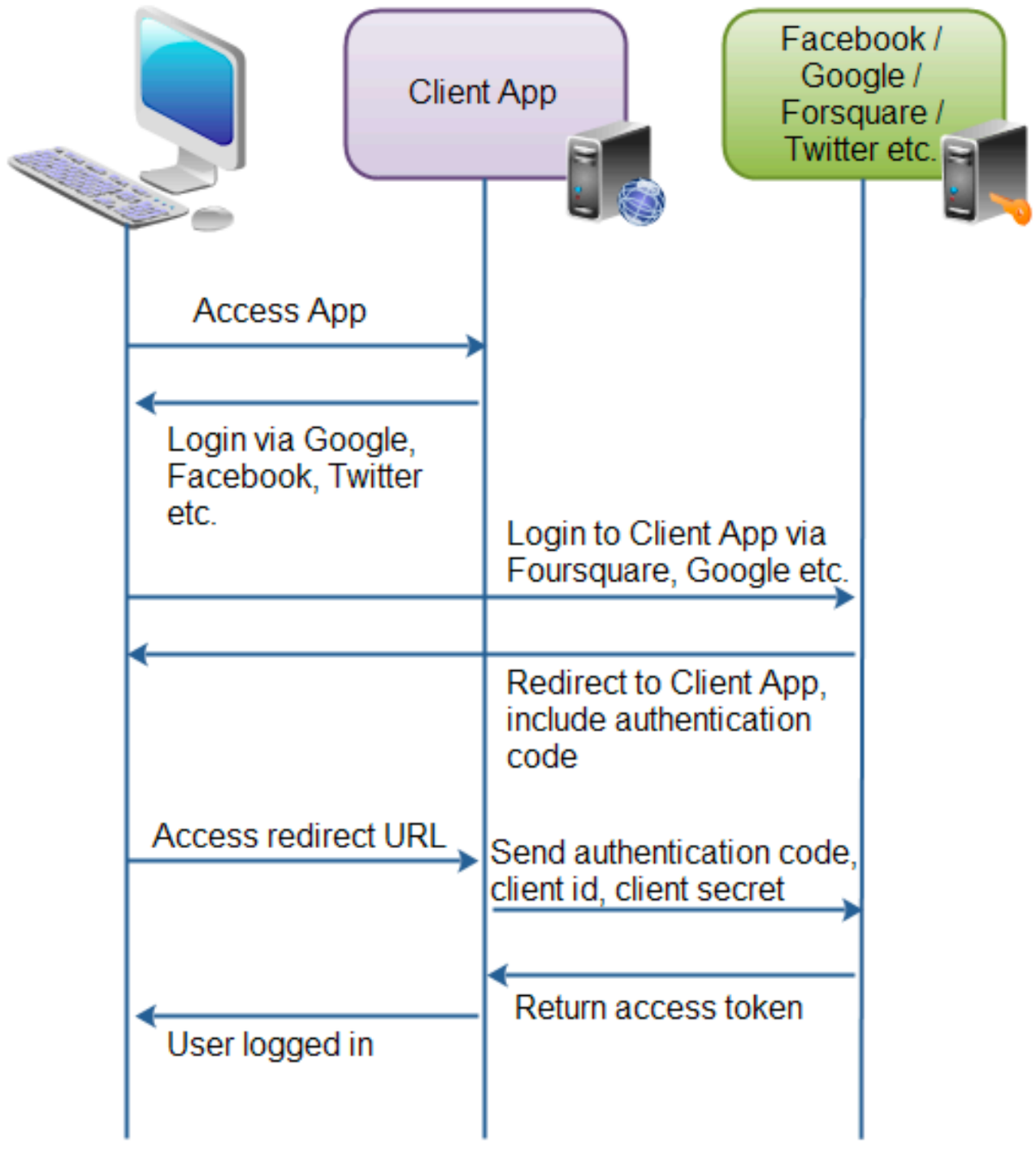
- Ganska simpelt och vanligt sätt att hantera åtkomst.
- Varje användare / konto får en ”nyckel”, en unik sträng, som används i varje request, antingen i URL eller i header.

- http://jsfiddle.net/emmio_micke/p5ntq1ro/

Oauth2

- Ett protokoll för att låta applikationer komma åt varandras data.

- Vi går inte in mer på den just nu, dock bra att ha hört talas om den.



CRUD

- Create, Read, Update, Delete
- Våra API-anrop hittills är Read-anrop där vi använder http-metoden `GET`.
- I vissa API:er kan man även göra de andra händelserna, oftast kopplat till behörighet.

HTTP verb	CRUD
POST	Create
GET	Read
PUT	Update/ Replace
DELETE	Delete

CRUD

- Det finns test-api:er man kan använda för att testa att skapa eller uppdatera data också.
- https://jsonplaceholder.typicode.com/
- Låt oss testa lite i Postman!

Resources

JSONPlaceholder comes with a set of 6 common resources:

/posts	100 posts
/comments	500 comments
/albums	100 albums
/photos	5000 photos
/todos	200 todos
/users	10 users

Note: resources have relations. For example: posts have man
full list.

Routes

All HTTP methods are supported. You can use http or https f

GET	/posts
GET	/posts/1
GET	/posts/1/comments
GET	/comments?postId=1
POST	/posts
PUT	/posts/1
PATCH	/posts/1
DELETE	/posts/1

Note: see guide for usage examples.

Fetch

- Nu när vi vet lite mer vad vi behöver kan vi skapa kod.
- Som vi såg när vi kollade på Basic Authentication så kan man konfigurera fetch genom att ange ett objekt.
- Där kan vi t ex sätta metoden till POST.
  - http://jsfiddle.net/emmio_micke/hgaf2o0r/

Uppgift

- Skapa en liten att göra-applikation.
- Analysera API:et med hjälp av dokumentationen och Postman.
  - https://jsonplaceholder.typicode.com/todos
  - https://jsonplaceholder.typicode.com/guide/
- Skapa ett enkelt gränssnitt och implementera de olika metoderna med fetch.
  - Ni kan säkert använda en del från den andra att göra-listan vi gjorde: https://yh.pingpong.se/courseId/11861/content.do?id=5209549