1. Goal

Learn to design the URI space of your API well.

This has traditionally been a stumbling point in API design, so we'll dig deep into this one in order to understand what each of these choices will mean mean for the API.

2. Lesson Notes

2.1. Nouns vs Verbs (or how to model actions)

Our goal for the URI structure of the API is to make it logical, meaningful and predictable for clients.

A foundational aspect in API design is that the API is Resource-centric. Operations (verbs) are represented via HTTP Verbs on these Resources, **not in the URI space**.

Rule of thumb: **a verb in the URI is a design smell**and always a sign that there's more domain modeling to be done.

Example:
- bad, using a verb in the URI:

```
/account/pay
```

- good, creating **a new Resource** for the payment concept:

```
/account/payment
```

In addition - DO NOT:

- use verbs in URI parameters
- use verbs that match the HTTP verbs: */getUser, /postRole/*

2.2. Plural vs Singular

You can chose either the singular or the plural form to represent the resources of the API - neither is wrong. Just keep in mind that it should be a consistent decision across the entire API.

Generally, plurals are more widely accepted in practice. Let's look at the **semantics of plurals**:

- with */privileges* - we're identifying a collection resource
- when we create (POST), we're adding to the collection
- when we're identifying a singular resource, we're drilling into the collection

2.3. IDs vs UUIDs?

The next decision is using either raw IDs or UUIDs in the URI space of the API.

CONS of IDs

- IDs are less flexible than UUIDs

- **ex**: if IDs / primary keys are used in the URI - then we're exposing aspects of the underlying technology we're using => which makes migrations harder to do

- IDSs are guessable

PROS of IDs

- IDs are small whereas UUIDs can add quite a lot to the payload of the message

- generating UUIDs is extra work, whereas ids are very likely already available

2.4. Representation Info in the URI

URIs should not contain any information about the representation:

The No No: *http://api.example.com/users/27.json*

The Evil Twin: *http://api.example.com/users/27?format=json*

The right way to ask a different Representation from the API is **to use the *Accept* (and Accept-Language) headers**.

3. Resources

- RESTful URI design