


- 08- Programming Foundations: APIs and Web Services

🔗 Certificate	Abschlusszertifikat_Programming Foundations APIs and Web Services.pdf
# Completed Sections	6
🔗 Course Links	https://www.linkedin.com/learning/programming-foundations-apis-and-web-services?resume=false
# Goal Sections	6
☰ Quick note	https://www.linkedin.com/learning/certificates/b75134954cb411b9993f1f867a1795cc8f0a6968dfa4ab789f33a1rk=share_certificate
📅 Start and Finish Date	
🖼️ Thumbnail	
⌵ Time	1h 14m 28s
⌵ Understanding %	
Σ نسبة الإنتهاء	✅ Done

Understanding Webservers

▼ Web services overview

- is an interaction between the client and the webserver
 - The client sends a message **and waits for a response from the server.**
The server receives the message, and the web service performs an action. Then, a message is sent back to the client.
- There are two types of webservers
 - SOAP
 - Sends messages using XML
 - XML document is sent with data
 - Rest
 - uses a web protocol HTTP, to access resources like documents , pic , or videos.

▼ Advantages of web services

- Reusability
 - It can be used by multiple systems. and that saves time
- language Transparency
 - no matter what is the language it can be used.
- Usability
 - It's an easy way to make the data more accessible to other systems in a secure fashion. The data can be used by a wide range of audiences and platforms
- Deployability



- They are deployed over standard internet technologies making them easily available on a global level.

▼ Considerations of web services

- Latency
 - The time it takes a request to return a response
 - if we rely on a lot of apis you will risk that it can take a lot of time to load
 - Low latency is good
 - High latency is not good



- Partial Failure
 - When a server or network fails to respond, because an action could not be taken



▼ Secure web services

1. Authentication

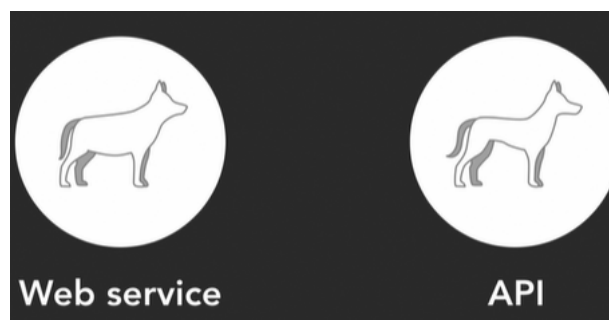
- validate identity of client
 - Typically, identity is validated with user credentials, such as a username and password.
 - Authentication is like simply unlocking your phone with your passcode. This step says yes, I have access to what's on this phone.
- Basic Authentication
 - The simplest protocol available for performing web services authentication over the HTTP protocol
- API key Authentication
 - Requires the API to be accessed with a unique key

2. Authorization

- is the next step after authentication
 - Determines level of client's access
 - What do they have access to
 - So, once a client is authenticated, they have proven who they are, what do they have access to? For example, what data can they view? Are they allowed to change that data, et cetera.

▼ Web services, APIs, and microservices

- All webservices are APIs, but not all APIs are web services.



- Web services

- Dependent on SOAP protocol
- requires more work to pack and unpack data
- API :: Microservices



- How to choose?
 - The business problem you're trying to solve?
 - The type of application you're trying to build?
 - and the capabilities of your calling clients.

Using RESTful APIs and HATEOAS

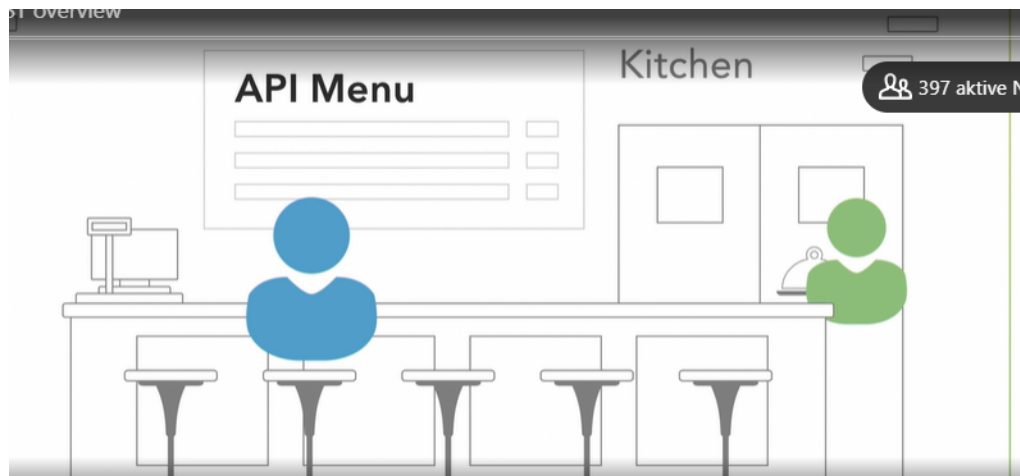
▼ REST Overview



- Representational State Transfer
 - A set of guidelines used to design APIs
 - APIs Principles
 1. Uniform resource Identifier(URL), the API are considered resources and identified through something called the URI,
 2. Operations
 - a. GET-Retrieves a resource
 - b. Post-Creates a resource
 - c. Put- Updates a resource
 - d. Delete- Removes a resource
 3. Formats
 - a. HTML
 - b. XML
 - c. Plain text
 4. Stateless

- a. Server will not store any state the client made. (it forget what you ordered like (sturbucks if you ordere something and than ask again about it.

- Example



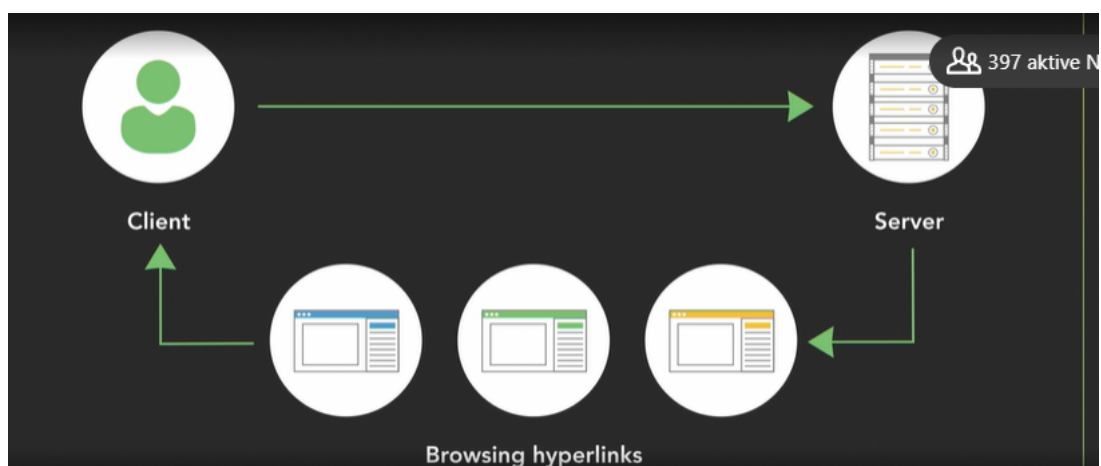
- Resources can not be manipulated using an unlimited set of operations.

▼ Benefits of REST

- Payload
 - Data sent between client and server
- It promotes loosely coupling
 - if you need to upgrade your API to provide new features, everyone using your API can keep using the same code without having **to upgrade their request to match**
 - RESTful APIs promote loose-coupling. This simply means that the system should be designed so that changes and enhancements to web services don't break clients that are already using them.
- it can starts small and evolve over time as new features are added

▼ HASTEOAS overview

- Client interacts with a REST API entirely through the responses by the server



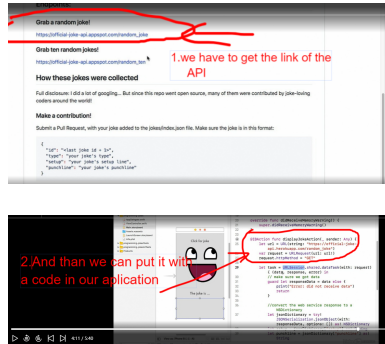
- Principle
 - Resources should be discoverable through the publication of links
- The primary advantage of HATEOAS is to avoid sending fields that require the client to interpret them and then decide what actions can be taken next. Instead, the server determines this ahead of time **and conveys what the client can**

do by the presence

or absence of the links provided.

- HATEOAS does not supports sending boolean fields along with state-related information.
- The primary advantage of HATEOAS is to avoid sending boolean fields or state-related fields that require the client to interpret them and decide what action(s) can be taken next.

▼ consume a RESTful API



▼ consume a RESTful API via Postman

- is a tool to test api with a better interface than in chrome.
- We save the Request into Collection



▼ Create a RESTful API

- if you want to share information with others: than use api
- Spring Data REST
 - Spring Data REST is a part of the umbrella Spring Data project and makes it easy **to build hypermedia driven RESTful web services.**
 - how to use
 - notice on lines 66-69 I've included the Spring DATA rest dependency and that's all **I need to do to start using Spring Data REST.**

▼ Document an API

- Swagger is a tool to document api
 - When creating a REST API, good documentation is instrumental and keeping that documentation updated with every new change is also instrumental.
 - Swagger Editor
 - we can write api specs
 - Swagger UI
 - So this renders OpenAPI specs as interactive API documentation.



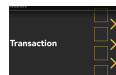
◦

Using SOAP- Based Web Services

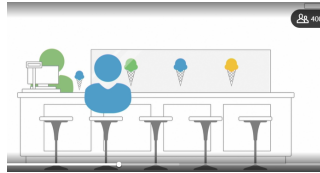
▼ SOAP overview



- Simple Object Access Protocol
 - SOAP can carry a bit more overhead. It allows for additional security, different kinds of transactions, and ACID compliance
- ▼ ACID and transactions
 - the transaction have to be very accurate.
 - A transaction is a set of operations that must all be completed, and if for some reason any of the individual operations aren't completed, **no changes are made to the database.**



- They have to be ACID
 - Atomic
 - there are indivisible, that pieces of it can't be separated out.
 - Consistent
 - it means that whatever the transaction does, it needs to leave the database in a valid or consistent state. The actions in a transaction can't violate integrity rules that are defined for the database.
 - Isolation
 - means that while the activities in the transaction are being completed, nothing else can make changes to the data involved.
 - Durability
 - means that the information we change in the transaction is reported as complete, the data is there.



- Proper Message Format have to be right
 - WSDL (Web service Description Language)
 - Contains information , like the data types being used in SOAP messages, and operations available via the web service.
 - SOAP Messages
 - Envelope
 - it's the starting and ending tags of the message.
 - Header
 - The header is optional. It contains the attributes of the message. We can add security tokens.
 - The Body
 - The body is required. It contains the actual XML data that the server transmits to the receiver.
 - Fault
 - And last, the fault, which is optional. The fault carries information about any errors that might occur during processing the message.


▼ History and future of SOAP

- it's complicated to use and that makes it good for security

SOAP

400 aktive Nutzerinnen

- **Used for enterprise-level web services**
 - Financial services
 - Payment gateways
 - CRM software
 - Identity management
 - Telecommunication services



- like pay pal

▼ Consume a SOAP web service

▼ SoapUI

- is a test tool to test soap api



▼ Create a web service

Developing APIs Using GraphQL

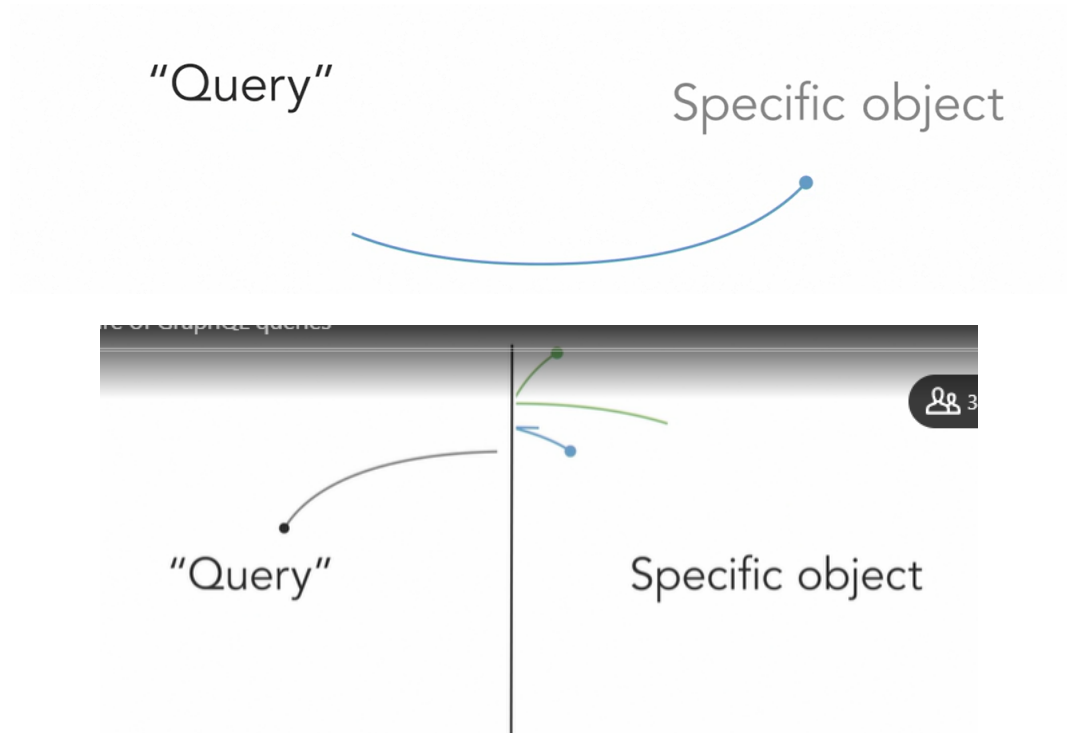
▼ GraphQL overview



- is query language for APIs. GraphQL is like having a personal assistant that handles all of the stops for you
- DEF
 - A syntax describing how to ask for data , which is usually used to load data.
 - Unlike regular SOAP or REST APIs, GraphQL gives you the ultimate flexibility in being able to specify in your API request specifically what data you need **and get back exactly that**.

▼ The structure of GraphQL queries

- is A string sent to a server to process and request data
- are read- only operations, and cannot be manipulated
- GraphQL queries are quite flexible, because they allow clients to get back only the data they need.
 - GraphQL Type System
 - Schema
 - defines a set of types
 - Queries
 - obtain information about specific fields from objects.
 - Resolvers
 - retrieves the data for us
- Field names
 - Unit of data (can be nested)
- Arguments
 - Set of key-value pairs (set selection)



- Data can be changed using GraphQL.
 - Besides query, there are two other operation types: mutation, which modifies server-side data, and subscription, which allows for notification of changes to data in realtime.
 - Mutation
 - modifies server-side data.
 - Subscription
 - notifies changes in data in real time.

▼ consume a GraphQL API



▼ Create an API with Graph QL