# OpenID Connect

## End-user Identity for Apps and APIs

Matthias Biehl

OpenID Connect


First edition: January 2019

Biehl, Matthias
API-University Press
Volume 6 of the API-University Series.
Includes illustrations, bibliographical references and index.
Built: 20190203105100
ISBN-13: 978-1979718479
ISBN-10: 1979718474


API-University Press
https://www.api-university.com
info@api-university.com

# Contents

# Abstract

Signup and login with a Google, Yahoo, or Microsoft account can be found in more and more web and mobile apps. One login used by many, freeing the end-user from the burden of managing many accounts and passwords. Signup and login to a new app become so smooth and convenient, that end-users are much more likely to try a new app.

For us developers of web and mobile apps, these signup and login features are attractive, too: we do not need to manage user credentials, and we get a higher conversion rate resulting in more new customers. In effect, this means cutting costs and increasing the number of new customers for our apps.

So how does this feature "Signup and login with Google, Yahoo, or Microsoft" work? It is realized with OpenID Connect, a standardized protocol for sharing end-user data in a secure and controlled manner. Exploring how OpenID Connect works, so we as developers can enjoy its benefits is the subject of this book.

This book explains the overall concept of OpenID Connect, so we understand who the actors are, which endpoints and tokens are involved and how these elements interact in so-called flows. These flows tend to get confusing, so we visualize these flows as sequence diagrams, and show how to choose the flow that is appropriate for a given scenario. Using examples, we explore how the tokens are constructed, signed and encrypted with JWT, JWS, and JWE.

This is not a programming book, don't expect implementations with a specific programming language or library. In-

stead, we focus on understanding OpenID Connect on a conceptual level, so we can design and architect apps that work with OpenID Connect. And OpenID Connect is the standard behind creating smooth login and signup experiences, increasing the customer signup rate, and creating highly converting apps.

# 1 Introduction

For us web and mobile app developers, authentication and identity management is a big pain point. We understand that it is critical from an information security perspective, since we are dealing with personal data, so we better get it right. This means setting up a secure infrastructure for authentication and storing user identity, maintaining it with constant security patches and providing end-user support, e.g. for lost passwords, changed address, changed email and a lot more. It simply means a lot of overhead and does not deliver any differentiating features to win new users. But signup and login are not only difficult to build and operate, but it is often inconvenient to use.

End-users are not very fond of filling in long signup forms, and if we require it in an app, it usually hurts conversion, number of signups, usage and ratings of the app. End-users tend to shy away from tedious onboarding processes or only get halfway through before they give up. Those users may never return.

So how do some of the most successful and popular apps deal with this situation? Have they found a solution, which may be the secret of their success?

To find out, let's study the user experience when signing up for their app. When signing up as a user for such an app, we can choose to identify with our existing Google, Yahoo, or Microsoft account. Chances are, we are already logged in to this account and a session is open, so we do not even need to provide my password.

As a result: Signup for the app is a smooth and convenient process, we just click "Signup with Google, Yahoo or Microsoft",

consent to the data sharing, and we are ready to use the app - our user account is created automatically.

This smooth process is not only used when signing up for the first time, but it is equally convenient when logging in later on.

For me as an end-user, this is so attractive, since these account credentials are the only ones I need to remember. I do not need to fill in another signup form, I do not need to create a new username and I do not need to remember another password. I can just use my existing account that I use every day. No wonder users love it.

So how do these apps achieve this convenient and smooth user experience for signup and login?

They use OpenID Connect. So let's explore what it is and how to use it.

## 1.1 What is OpenID Connect

OpenID Connect (OIDC) is a protocol for providing identity as a service. It is a technical protocol, that allows end-user data (so-called claims or attributes) to be passed in a secure manner from a provider to a client, with the explicit consent of the end-user. The protocol can ensure the integrity and confidentiality of the user data. Thus, OpenID Connect allows app developers to have their users authenticate with the OpenID Connect provider ("as a service"), and receive the claims and attributes of the authenticated end-user in a secure, integrity-protected way. The client/app can use these claims and attributes internally to provision an account that is linked to the user's resources in the client.

## 1.2 How does OpenID Connect Work

OpenID Connect [10] is a simple, standardized identity layer, which is realized on top of the popular OAuth 2.0 protocol [4]. All the concepts, flows, endpoints and tokens of OAuth 2.0 also apply in the context of OpenID Connect. In addition, OpenID Connect can also deliver claims (a.k.a. attributes) of the end-user. OpenID Connect delivers claims of the end-user in two ways to an app:

- via a RESTful userinfo API, which is protected with OAuth (see section 1.2.1), and
- via a standardized identity token, which conforms to the JWT standard (see section 1.2.2).

### 1.2.1 Claims on the Userinfo Endpoint

The RESTful API, which provides the claims, is called userinfo endpoint (see section 3.3). The API is protected by OAuth, which means that a valid OAuth access token needs to be provided when the API is accessed. In addition, the correct OAuth scopes need to be set for the access token. In the OAuth terminology, the term scope is used to refer to a specific access right.

The OpenID Connect standard specifies one obligatory scope, the `openid` scope and additional optional scopes such as `email` or `address`. The obligatory scope always needs to be present and delivers a basic set of claims. To receive anything more than this basic set of claims, optional scopes need to be set.

So how does the app obtain an OAuth token with specific scopes? First, the clientID used by the app needs to be configured to allow the respective scopes. Second, the app needs to request the respective scopes when obtaining the authorization_code. Third, the end-user (a.k.a. resource owner, see sec-

tion 2.3) needs to confirm all the scopes on the UI presented by the OAuth Consent Server.

### 1.2.2 Claims in the Identity Token

The second option how OpenID Connect delivers claims and attributes of the end-user is the identity token (see section 4.5). It is provided by the token endpoint (see section 3.4) along with the OAuth access token. The identity token is an alternative, standardized form of obtaining claims/attributes. In a typical interaction (Authorization Code Flow, see section 5.1), the identity token is created by the token endpoint and provided along with the access token in a JSON structure.

The identity token actually contains the claims in an encoded form. Unlike the OAuth access token which is just a random character sequence, the identity token actually contains the claims, i.e. the claims are encoded into the token. This design allows the third party app to access the claims offline. The third party app only needs the identity token and the cryptographic keys to validate the token.

The representation of an identity token is a relatively long string. In contrast to all other OAuth tokens, this token actually contains information. It is a JSON web token (JWT, see section 7.1). JSON web tokens are created by putting mandatory and optional claims into a JSON object, Base64URL-encoding this JSON object, and signing the resulting string using one of the cryptographic algorithms defined in the JWT standard. The signed token is integrity protected, meaning that the token cannot be manipulated without being detected.

## 1.3 OAuth 2 vs. OpenID Connect

For API security there are two standards — and both of their names start with the capital letter O. So it is no wonder, people

ask all the time: What is the difference between OAuth 2 and OpenID Connect?

The OAuth standard ensures that there is no unintended leakage of information about the resource owner to the app. The app may access only specific resources with the explicit consent of the resource owner. The app does not get the resource owner's credentials, which would be a wildcard access to all of the user's data. By protecting the resource owner's data, especially the personal and profile data of the resource owner, the OAuth standard can be used to ensure the privacy of the resource owner.

The strict privacy policy of OAuth is a good default setting. There are, however, cases in which the resource owner wants that the identity provider hands specific data of a resource owner, for example, the resource owner's name or address, to a specific app, i.e. for a smooth sign-up and login experience. Of course, the access right to this information is only provided, if the resource owner explicitly consents to the delegation of the respective access rights to the app.

OpenID Connect standardizes how apps can access the attributes of the resource owner via a token and via a RESTful API and how this data is structured and organized. OpenID Connect extends the authorization code flow, introduces new tokens and standardizes some endpoints. OpenID Connect is a solution that can be applied in many environments, on many devices, and with many different products. OpenID Connect is realized as an extension of OAuth, as a so-called OAuth profile. OAuth profiles are a standardized mechanism to build upon the main OAuth standard.

## 1.4 Common Misunderstandings

*"We already got OAuth, so we don't need OpenID Connect"*

Great, that you have OAuth. If you have it long enough, you probably have a big API portfolio and have very likely come across the need to expose customer data, employee data or partner data. So you wrote one or multiple APIs to deliver person attributes. The Person API - every company creates an API with data about the people it deals with. Sometimes this API is called a customer API, an employee API, a user API or a me API. This API delivers data about a person – not about any person – but just about the person that is currently logged in. This API delivers identity information, including the name, the email address and a profile picture. One aspect of OpenID Connect is the attempt to standardize this kind of API. In OpenID Connect this standardized API is called userinfo API.

*"OpenID Connect is an authentication standard"*

No. OpenID Connect is agnostic to the authentication mechanism you use. It does not prescribe how you have to authenticate a user. But having an authentication mechanism - any authentication mechanism in place, is a precondition for using OpenID Connect.

*"OpenID is a short form for OpenID Connect"*

No. OpenID and OpenID Connect are separate standards - maintained by the same foundation and addressing a similar set of problems. OpenID is an older standard using proprietary technologies, so it is difficult to implement, often error-prone and incompatible. For this reason, it never found widespread acceptance. OpenID Connect is built on many web standards and best practices such as OAuth, JSON, TLS, JWT, JWS, JWE, JWK, JWA.

## 1.5 Perspective of the End-user

Let me give you an example: Tim signs up for a new SaaS application tweetbuffer.com to automate his social media channels.

When signing up for an account on tweetbuffer.com, he does not need to enter a new username, choose a password, and enter all of his profile data. He can skip all that and simply choose the Google OpenID Connect provider.

When signing up on tweetbuffer.com (client) with the Google OpenID Connect provider, Tim (user) gets redirected to the accounts.google.com login page (OpenID Connect Provider - authentication component), where he logs in with existing Google credentials. He then needs to confirm that tweetbuffer.com can access his account and profile details (OpenID Connect Provider - consent management component). Under the hood, without Tim noticing, tweetbuffer.com will receive an OAuth access token and an OpenID Connect ID token.

With the access token, tweetbuffer.com can call the userinfo endpoint of Google to receive profile information and provision the user information to tweetbuffer.

### Advantages for the End-user

Advantages of the OpenID Connect solution for the end-user are:

- Experience smooth signup and login
- No need to fill in signup forms
- No need to sign in if a session with the OpenID Connect Provider already exists
- Not having to remember another username/password

## 1.6 Perspective of the App Provider

The App provider who wants to use Open ID Connect needs to follow these steps:

- Register the client/app with the OpenID Connect Provider. Get the credentials and implement the redirect endpoint.

- Write an app, and when the end-users want to sign up or login, redirect them to the login page of the OpenID Connect Provider. Then, ask the OpenID Connect Provider to give you an OAuth Access Token and an OpenID Connect ID Token.

- In the app, access the end-user identity and profile information. This can be done in two ways: via the claims in the OpenID Connect ID Token or by calling an userinfo endpoint, an OAuth protected API, that provides claims about the end-user.

### Advantages for the App Provider

Using OpenID Connect in the App brings the following advantages for the App provider:

- Providing a smooth signup and login experience for the end-users, resulting in higher conversion rates, more signups, and higher retention rates.

- Not having to deal with identity management and authentication infrastructure, resulting in lower costs.

- Not having to set up a support organization for identity-related issues, such as lost passwords, locked accounts, changed address, new email address or new name, resulting in lower costs.

- Lower security risk: Not having to manage any user credentials means that there is no risk of losing them.

## 1.7 Perspective of the OpenID Connect Provider

OpenID Connect helps identity providers to deliver identity information in a trustworthy, secure and standardized way. Right now, the big technology players (e.g. Google, Yahoo, Microsoft) are OpenID Connect providers for the general public, meaning that any end-users can open accounts with these companies and OpenID Connect is automatically enabled for these accounts.

But there is more space on this market: governments and local institutions that already today are the trusted source of identity, are also well-positioned to become OpenID Connect providers for the general public.

But OpenID Connect providers do not necessarily need to address the general public. OpenID Connect providers can also address closed user circles, such as the employees of a company. Every company can become an OpenID Connect provider for their employees.

So there is an opportunity to establish providers of user identity and bind customers to that service. For the providers, this is an opportunity to build an ecosystem or expand an existing ecosystem. Those companies who become OpenID Connect providers, create a presence with their brand on many login screens of exactly those third-party apps, which their customers are interested in. And those apps will form the ecosystem. When end-users make use of their OpenID Connect provider's service to connect to other apps and services, they actually strengthen their ties to the OpenID Connect provider and are unlikely to change service.

The end-users leave traces of their activity with the OpenID Connect provider. The end-users trust the OpenID Connect provider to act responsibly with this data. There is certainly an opportunity for the OpenID Connect provider to analyze this

data to improve its services and offerings.

### Advantages for the OpenID Connect Provider

The OpenID Connect provider creates an infrastructure that is often not directly monetized. There are the following advantages for the OpenID Connect Provider, which can improve its strategic position in the mid to long term:

- Opportunity to build an ecosystem of partners (app developers) serving the same customers.
- Opportunity to establish and reinforce customer's trust and partner's trust.
- Bind end-users to the OpenID Connect Provider and all of its services, by becoming a central player in the digital ecosystem of the customer.
- Opportunity for branding by being present on many login screens of your customer's apps. Can be seen as a marketing channel.
- Opportunity to build a database of customers and potential prospects for other services besides OpenID Connect.
- Opportunity to learn responsibly about end-users, their apps, and their usage patterns of the partner's apps.
- Use the collected data to create insights about customers and products, improve offerings, services, and sales.

## 1.8 OpenID Connect Cheat Sheet

Here is a free OpenID Connect Cheat Sheet, for a quick overview of OpenID Connect. Download it, print it, keep it on the desk

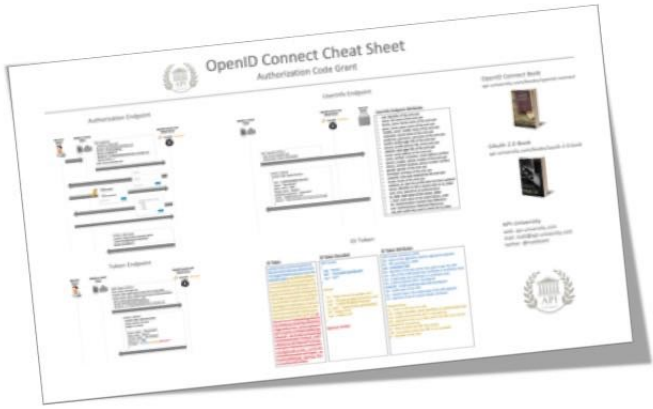


Figure 1.1: OpenID Connect Cheat Sheet

and never get confused by OpenID Connect again: https://api-university.com/openid-connect-cheat-sheet

# Appendix

## Feedback

If you enjoyed this book and got some value from it, it would be great if you could share with others what you liked about the book on the Amazon review page.

If you feel something was missing or you are not satisfied with your purchase, please contact me at matt@api-university.com. I read this email personally and am very interested in your feedback.

## About the Author

Matthias has provided expertise to international and national companies in software architecture, software development processes, and software integration. At some point, he got a Ph.D.

Nowadays, Matthias uses his background in software engineering to help companies to realize their digital transformation agenda and to bring innovative software solutions to the market.

He also loves sharing his knowledge in the classroom, at workshops, and in his books. Matthias is an instructor at the API-University, publishes a blog on APIs, is an author of several books on APIs and regularly speaks at technology conferences.

# Other Products by the Author

## Book on RESTful API Design

Looking for Best Practices in RESTful APIs?

This book is for you! Why? Because this book is packed with best practices on many technical aspects of RESTful API Design, such as the correct use of resources, URIs, representations, content types, data formats, parameters, HTTP status codes, and HTTP methods.

You want to design and develop APIs like a Pro? Use API description languages to both design APIs and develop APIs efficiently. The book introduces the two most common API description languages RAML and OpenAPI/Swagger.

Your APIs connect to legacy systems? The book shows best practices for connecting APIs to existing backend systems.

You expect lots of traffic on your API? The book shows you how to achieve high security, performance, availability and smooth evolution and versioning.

Your company cares about its customers? Learn a customer-centric design and development approach for APIs, so you can design APIs as digital products.

Title: RESTful API Design
Author: Matthias Biehl
Release Date: 2016-08-30
Length: 290 pages
ISBN-13: 978-1514735169

https://api-university.com/books/api-design

## Book on GraphQL API Design

Want to build APIs like Facebook? Since Facebook's framework for building APIs, GraphQL, has become publicly available, this ambition seems to be within reach for many companies. And that is great. But first, let's learn what GraphQL really is and – maybe even more importantly – let's figure out how to apply GraphQL to build APIs that consumers love.

In this book, we take a hands-on approach to learning GraphQL. We first explore the concepts of the two GraphQL languages using examples. Then we start writing some code for our first GraphQL API. We develop this API step by step, from creating a schema and resolving queries, over mocking data and connecting data sources all the way to developing mutations and setting up event subscriptions.

Are your API consumers important to you? This book shows you how to apply a consumer-oriented design process for GraphQL APIs.

Do you want to enable the API consumers so they can build great apps? This book explains the GraphQL query language, which allows the API consumers to retrieve data, write data and get notified when data changes.

Do you want to make your API easy and intuitive to use? This book shows you how to use the GraphQL schema language to define a type system for your API.

Title: GraphQL API Design
Author: Matthias Biehl
Release Date: 2017-12-30
Length: 95 pages
ISBN-13: 978-1979717526

https://api-university.com/books/graphql-api-design

## Book on Webhooks

Got a RESTful API? Great. API consumers love them. But today, such RESTful APIs are not enough for the evolving expectations of API consumers. Their apps need to be responsive, event-based and react to changes in near real-time.

This results in a new set of requirements for the APIs, which power the apps. APIs now need to provide concepts such as events, notifications, triggers, and subscriptions. These concepts are not natively supported by the REST architectural style.

The good thing is that we can engineer RESTful APIs that support events with a webhook infrastructure. But it requires some heavy lifting. The webhook infrastructure needs to be developer-friendly, easy to use, reliable, secure and highly available.

With the best practices and design templates provided in this book, we want to help you extend your API portfolio with a modern webhook infrastructure. So you can offer both APIs and events that developers actually love to use.

Title: Webhooks - Events for REST APIs
Author: Matthias Biehl
Release Date: 2017-12-30
Length: 120 pages
ISBN-13: 978-1979717069

https://api-university.com/books/webhooks

## Book on API Architecture

Looking for the big picture of building APIs? This book is for you!

Building APIs that consumers love should certainly be the goal of any API initiative. However, it is easier said than done. It requires getting the architecture for your APIs right. This book equips you with both foundations and best practices for API architecture. This book presents best practices for putting an infrastructure in place that enables efficient development of APIs. This book is for you if you want to understand the big picture of API design and development, you want to define an API architecture, establish a platform for APIs or simply want to build APIs your consumers love. What is API architecture? Architecture spans the bigger picture of APIs and can be seen from several perspectives: The architecture of the complete solution, the technical architecture of the API platform, the architecture of the API portfolio, the design decisions for a particular API proxy. This book covers all of the above perspectives on API architecture. However, to become useful, the architecture needs to be put into practice. This is why this book covers an API methodology for design and development. An API methodology provides practical guidelines for putting API architecture into practice. It explains how to develop an API architecture into an API that consumers love.

Title: API Architecture
Author: Matthias Biehl
Release Date: 2015-05-22
Length: 190 pages
ISBN-13: 978-1508676645

https://api-university.com/books/api-architecture

## Book on OAuth 2.0

This book offers an introduction to API Security with OAuth 2.0. In less than 80 pages you will gain an overview of the capabilities of OAuth. You will learn the core concepts of OAuth. You will get to know all 4 OAuth Flows that are used in cloud solutions and mobile apps. If you have tried to read the official OAuth specification, you may get the impression that OAuth is complicated. This book explains OAuth in simple terms. The different OAuth Flows are visualized graphically using sequence diagrams. The diagrams allow you to see the big picture of the various OAuth interactions. This high-level overview is complemented with a rich set of example requests and responses and an explanation of the technical details. In the book the challenges and benefits of OAuth are presented, followed by an explanation of the technical concepts of OAuth. The technical concepts include the actors, endpoints, tokens and the four OAuth flows. Each flow is described in detail, including the use cases for each flow. Extensions of OAuth - so called profiles - are presented, such as OpenID Connect and the SAML2 Bearer Profile. Sequence diagrams are presented to explain the necessary interactions.

Title: OAuth 2.0 - Getting Started in Web-API Security
Author: Matthias Biehl
Release Date: 2014-11-15
Length: 76 pages
ISBN-13: 978-1507800911

https://api-university.com/books/oauth-2-0-book

## Online Course on OAuth 2.0

Securing APIs is complicated? This course offers an introduction to API Security with OAuth 2.0. In 3 hours you will gain an overview of the capabilities of OAuth. You will learn the core concepts of OAuth. You will get to know all 4 OAuth flows that are used in cloud solutions and mobile apps. You will also be able to look over the shoulder of an expert using OAuth for the APIs of Facebook, LinkedIn, Google and Paypal.

Title: OAuth 2.0 - Getting Started in Web-API Security
Lecturer: Matthias Biehl
Release Date: 2015-07-30
Material: Video, Workbooks, Quizzes
Length: 4h

https://api-university.com/courses/oauth-2-0-course

## Online Course on RESTful API Design

Looking for best practices of RESTful API Design? This course is for you! Why? This course provides interactive video tutorials on the best practices of RESTful design. These best practices are based on the lessons learned from building and designing APIs over many years.

The course also includes video lectures on technical aspects of RESTful API Design, including the correct use of resources, URIs, representations, content-types, data formats, parameters, HTTP status codes and HTTP methods. And thanks to many interactive quizzes, learning REST becomes an engaging and exciting game-like experience.

We focus on the practical application of the knowledge, to get you ready for your first RESTful API project. The course includes guided mini-projects to get you ready for the practical application of REST.

After completing this course, you will be able to design RESTful APIs – but not just any APIs, you have all the knowledge to design APIs, which your consumers will love.

Title: RESTful API Design
Lecturer: Matthias Biehl
Release Date: 2019-06-01
Material: Video, Workbooks, Quizzes
Length: 3h

https://api-university.com/courses/restful-api-design-course

# Bibliography


[1] Matthias Biehl. *RESTful API Design: Best Practices in API Design with REST (API-University Series Book 3)*. 1 edition, August 2016. 3

[2] John Bradley, Nat Sakimura, and Michael Jones. JSON web signature (JWS). Technical Report 7515, RFC Editor, Fremont, CA, USA, May 2015. 7.2

[3] John Bradley, Nat Sakimura, and Michael Jones. JSON web token (JWT). Technical Report 7519, RFC Editor, Fremont, CA, USA, May 2015. 7.1

[4] Dick Hardt. The OAuth 2.0 authorization framework. Technical Report 6749, RFC Editor, Fremont, CA, USA, October 2012. 1.2

[5] Joe Hildebrand and Michael Jones. JSON web encryption (JWE). Technical Report 7516, RFC Editor, Fremont, CA, USA, May 2015. 7.3, 7.3.2.1

[6] IANA. MIME Media Types, March 2007. 7.2.1.1, 7.3.1.1

[7] Leif Johansson. An IANA registry for level of assurance (LoA) profiles. Technical Report 6711, RFC Editor, Fremont, CA, USA, August 2012. 3.3.3, 4.5.2.1

[8] Michael Jones. JSON web algorithms (JWA). Technical Report 7518, RFC Editor, Fremont, CA, USA, May 2015. 7.2.1.1, 7.3.1.1, 7.5

[9] Michael Jones. JSON web key (JWK). Technical Report 7517, RFC Editor, Fremont, CA, USA, May 2015. 7.4

[10] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, and C. Mortimore. OpenID connect core 1.0. Technical report, OpenID Foundation, November 2014. 1.2

[11] Robert W. Shirey. Internet security glossary, version 2. Technical Report 4949, RFC Editor, Fremont, CA, USA, August 2007. 7

[12] Anne van Kesteren. Cross-Origin resource sharing (CORS), March 2009. 3.3.2

# Index