

Curriculum für

Certified Professional for
Software Architecture (CPSA)[®]
Advanced Level

**Modul
API**

Application Programming Interfaces

2025.1-rev0-DE-20250306



Inhaltsverzeichnis

Verzeichnis der Lernziele	2
Einführung: Allgemeines zum iSAQB Advanced Level	3
Was vermittelt ein Advanced Level Modul?	3
Was können Absolventen des Advanced Level (CPSA-A)?	3
Voraussetzungen zur CPSA-A-Zertifizierung	3
Grundlegendes	4
Was vermittelt das Modul „API“?	4
Struktur des Lehrplans und empfohlene zeitliche Aufteilung	4
Dauer, Didaktik und weitere Details	4
Voraussetzungen	5
Gliederung des Lehrplans	5
Ergänzende Informationen, Begriffe, Übersetzungen	5
1. Warum APIs wichtig sind	6
1.1. Begriffe und Konzepte	6
1.2. Lernziele	6
1.3. Referenzen	6
2. Wie APIs Wert erzeugen	7
2.1. Begriffe und Konzepte	7
2.2. Lernziele	7
2.3. Referenzen	7
3. API-Stile und -Technologien	8
3.1. Begriffe und Konzepte	8
3.2. Lernziele	8
3.3. Referenzen	8
4. API-Design	9
4.1. Begriffe und Konzepte	9
4.2. Lernziele	9
4.3. Referenzen	10
5. Beschreibung von APIs	11
5.1. Begriffe und Konzepte	11
5.2. Lernziele	11
5.3. Referenzen	11
6. API-Lifecycle und API-Tooling	12
6.1. Begriffe und Konzepte	12
6.2. Lernziele	12
6.3. Referenzen	12
7. API-Security	13



7.1. Begriffe und Konzepte 13

7.2. Lernziele 13

7.3. Referenzen 13

8. APIs at Scale: Plattformen und Governance 14

8.1. Begriffe und Konzepte 14

8.2. Lernziele 14

8.3. Referenzen 14

Referenzen 15

© (Copyright), International Software Architecture Qualification Board e. V. (iSAQB® e. V.) 2025

Die Nutzung des Lehrplans ist nur unter den nachfolgenden Voraussetzungen erlaubt:

1. Sie möchten das Zertifikat zum CPSA Certified Professional for Software Architecture Foundation Level® oder CPSA Certified Professional for Software Architecture Advanced Level® erwerben. Für den Erwerb des Zertifikats ist es gestattet, die Text-Dokumente und/oder Lehrpläne zu nutzen, indem eine Arbeitskopie für den eigenen Rechner erstellt wird. Soll eine darüber hinausgehende Nutzung der Dokumente und/oder Lehrpläne erfolgen, zum Beispiel zur Weiterverbreitung an Dritte, Werbung etc., bitte unter info@isaqb.org nachfragen. Es müsste dann ein eigener Lizenzvertrag geschlossen werden.
2. Sind Sie Trainer oder Trainingsprovider, ist die Nutzung der Dokumente und/oder Lehrpläne nach Erwerb einer Nutzungslizenz möglich. Hierzu bitte unter info@isaqb.org nachfragen. Lizenzverträge, die alles umfassend regeln, sind vorhanden.
3. Falls Sie weder unter die Kategorie 1. noch unter die Kategorie 2. fallen, aber dennoch die Dokumente und/oder Lehrpläne nutzen möchten, nehmen Sie bitte ebenfalls Kontakt unter info@isaqb.org zum iSAQB e. V. auf. Sie werden dort über die Möglichkeit des Erwerbs entsprechender Lizenzen im Rahmen der vorhandenen Lizenzverträge informiert und können die gewünschten Nutzungsgenehmigungen erhalten.

Wichtiger Hinweis

Grundsätzlich weisen wir darauf hin, dass dieser Lehrplan urheberrechtlich geschützt ist. Alle Rechte an diesen Copyrights stehen ausschließlich dem International Software Architecture Qualification Board e. V. (iSAQB® e. V.) zu.

Die Abkürzung "e. V." ist Teil des offiziellen Namens des iSAQB und steht für "eingetragener Verein", der seinen Status als juristische Person nach deutschem Recht beschreibt. Der Einfachheit halber wird iSAQB e. V. im Folgenden ohne die Verwendung dieser Abkürzung als iSAQB bezeichnet.

Verzeichnis der Lernziele

- LZ 1-1: APIs in die Historie der Softwareentwicklung einordnen
- LZ 1-2: Unterschiedliche Integrationsstile und -konzepte vergleichen
- LZ 2-1: Das Konzept der "API Economy" verstehen
- LZ 2-2: Verschiedene Arten der API-Wertschöpfung kennen
- LZ 2-3: API-Geschäftsmodelle verstehen
- LZ 2-4: APIs und digitale Transformation verstehen
- LZ 3-1: Grundlegende API-Stile verstehen
- LZ 3-2: Populäre API-Technologien einordnen und den jeweiligen API-Stilen zuordnen
- LZ 3-3: Auswahlkriterien und Konsequenzen von API-Stilen und API-Technologien kennen
- LZ 4-1: Bedeutung von API-Design verstehen
- LZ 4-2: API-Design als "Outside-in"-Ansatz anwenden
- LZ 4-3: Offenheit und Erweiterbarkeit als wichtige Aspekte für die Weiterentwicklung von APIs interpretieren
- LZ 4-4: Versionierung von APIs verstehen
- LZ 4-5: Good Practices für gelungenes API-Design kennen
- LZ 5-1: Wichtigkeit von API-Beschreibungen kennen
- LZ 5-2: OpenAPI als Beschreibungssprache für HTTP-APIs einordnen
- LZ 5-3: Den Aufbau von OpenAPI verstehen
- LZ 5-4: Alternative Beschreibungssprachen kennen
- LZ 6-1: API-Lifecycle verstehen
- LZ 6-2: APIs als Produkte verwalten
- LZ 6-3: API-Lifecycle-Tooling kennen
- LZ 7-1: Grundlagen von Kommunikationssicherheit kennen
- LZ 7-2: Relevante Technologien im API-Umfeld verstehen
- LZ 7-3: OWASP API Security Top 10 kennen
- LZ 8-1: Verschiedene Plattform-Begriffe vergleichen
- LZ 8-2: API-Guidelines aufstellen
- LZ 8-3: API-Guidelines by Example umsetzen
- LZ 8-4: APIs als Team-Schnittstellen sehen

Einführung: Allgemeines zum iSAQB Advanced Level

Was vermittelt ein Advanced Level Modul?

Das Modul kann unabhängig von einer CPSA-F-Zertifizierung besucht werden.

- Der iSAQB Advanced Level bietet eine modulare Ausbildung in drei Kompetenzbereichen mit flexibel gestaltbaren Ausbildungswegen. Er berücksichtigt individuelle Neigungen und Schwerpunkte.
- Die Zertifizierung erfolgt als Hausarbeit. Die Bewertung und mündliche Prüfung wird durch vom iSAQB benannte Expert:innen vorgenommen.

Was können Absolventen des Advanced Level (CPSA-A)?

CPSA-A-Absolventen können:

- eigenständig und methodisch fundiert mittlere bis große IT-Systeme entwerfen
- in IT-Systemen mittlerer bis hoher Kritikalität technische und inhaltliche Verantwortung übernehmen
- Maßnahmen zur Erreichung von Qualitätsanforderungen konzeptionieren, entwerfen und dokumentieren sowie Entwicklungsteams bei der Umsetzung dieser Maßnahmen begleiten
- architekturelevante Kommunikation in mittleren bis großen Entwicklungsteams steuern und durchführen

Voraussetzungen zur CPSA-A-Zertifizierung

- erfolgreiche Ausbildung und Zertifizierung zum Certified Professional for Software Architecture, Foundation Level® (CPSA-F)
- mindestens drei Jahre Vollzeit-Berufserfahrung in der IT-Branche; dabei Mitarbeit an Entwurf und Entwicklung von mindestens zwei unterschiedlichen IT-Systemen
 - Ausnahmen sind auf Antrag zulässig (etwa: Mitarbeit in Open-Source-Projekten)
- Aus- und Weiterbildung im Rahmen von iSAQB-Advanced-Level-Schulungen im Umfang von mindestens 70 Credit Points aus mindestens drei unterschiedlichen Kompetenzbereichen
- erfolgreiche Bearbeitung der CPSA-A-Zertifizierungsprüfung



Grundlegendes

Was vermittelt das Modul „API“?

Application Programming Interfaces, oder APIs, ermöglichen die Nutzung von digitalen Diensten über programmierbare Schnittstellen. Ihre Nutzung hat über die letzten 20 Jahre stark zugenommen. Aus Sicht der Softwarearchitektur sind APIs sowohl aus Produzenten- als auch aus Konsumentensicht relevant: Softwarekomponenten sollten idealerweise als wiederverwendbare Produkte verstanden werden; APIs helfen, eine Softwarekomponente einfach wiederverwendbar zu machen. Heute stehen mehr und mehr Dienste über APIs zur Verfügung; als Konsument dieser Dienste kann man sich besser auf Kernaufgaben konzentrieren und andere Aufgaben auslagern an externe Komponenten.

Das Modul betrachtet die verschiedenen Arten, wie APIs Wert erzeugen: Als technische Schnittstellen die es verschiedenen Komponenten ermöglichen miteinander zu kommunizieren; als organisatorische Schnittstellen die es verschiedenen Teams ermöglicht mit weniger gegenseitigen Abhängigkeiten ihre jeweiligen Komponenten zu entwickeln; und als geschäftsorientierte Bausteine die es der Organisation erlauben, neue Dienste schneller und effektiver zu entwickeln.

Im Rahmen dieses Moduls sprechen wir bei APIs immer von Netzwerk-basierten Schnittstellen, also nicht von lokalen Programmierschnittstellen.

Neben dem Schwerpunkt auf technischem Themen werden auch strategische Aspekte behandelt wie die Wertschöpfung und die Verwendungen von APIs als Team-Kollaboration im Unternehmen. Damit eignet sich das Modul gut, um sich dem Thema APIs mit einer weiteren Perspektive als dem rein technischen Blick zu nähern.

Struktur des Lehrplans und empfohlene zeitliche Aufteilung

Inhalt	Empfohlene Mindestdauer (min)	Übungszeit (Minuten)
1. Warum APIs wichtig sind	60	0
2. Wie APIs Wert erzeugen	60	30
3. API Stile und Technologien	60	30
4. API Design	90	30
5. Beschreibung von APIs	90	30
6. API Lifecycle und API Tooling	60	30
7. API Security	60	30
8. APIs at Scale: Plattformen und Governance	60	0
Summe	540 (9 h)	180 (3 h)

Dauer, Didaktik und weitere Details

Die unten genannten Zeiten sind Empfehlungen. Die Dauer einer Schulung zum Modul API sollte mindestens 2 Tage betragen, kann aber länger sein. Anbieter können sich durch Dauer, Didaktik, Art und Aufbau der Übungen sowie der detaillierten Kursgliederung voneinander unterscheiden. Insbesondere die Art der Beispiele und Übungen lässt der Lehrplan komplett offen.

Lizenzierte Schulungen zu API tragen zur Zulassung zur abschließenden Advanced-Level-

Zertifizierungsprüfung folgende Credit Points) bei:

Methodische Kompetenz:	10 Punkte
Technische Kompetenz:	10 Punkte
Kommunikative Kompetenz:	0 Punkte

Voraussetzungen

Teilnehmerinnen und Teilnehmer sollten folgende Kenntnisse und/oder Erfahrung mitbringen:

- Arbeit als Architekt:innen oder vergleichbare konzeptionelle oder Management Aufgaben.
- Arbeit als Entwickler:innen oder vergleichbare technische Aufgaben.
- Technisch orientierte Personen mit einem Basiswissen IT und Softwarearchitektur, die sich einen umfassenden Überblick über das Thema APIs verschaffen möchten.

Gliederung des Lehrplans

Die einzelnen Abschnitte des Lehrplans sind gemäß folgender Gliederung beschrieben:

- **Begriffe/Konzepte:** Wesentliche Kernbegriffe dieses Themas.
- **Unterrichts-/Übungszeit:** Legt die Unterrichts- und Übungszeit fest, die für dieses Thema bzw. dessen Übung in einer akkreditierten Schulung mindestens aufgewendet werden muss.
- **Lernziele:** Beschreibt die zu vermittelnden Inhalte inklusive ihrer Kernbegriffe und -konzepte.

Dieser Abschnitt skizziert damit auch die zu erwerbenden Kenntnisse in entsprechenden Schulungen.

Ergänzende Informationen, Begriffe, Übersetzungen

Soweit für das Verständnis des Lehrplans erforderlich, haben wir Fachbegriffe ins [iSAQB-Glossar](#) aufgenommen, definiert und bei Bedarf durch die Übersetzungen der Originalliteratur ergänzt.

1. Warum APIs wichtig sind

Dauer: 60 Min.	Übungszeit: 0 Min.
----------------	--------------------

1.1. Begriffe und Konzepte

Lokale APIs, netzwerkbasierte APIs, API Landschaft, Systemintegration

1.2. Lernziele

LZ 1-1: APIs in die Historie der Softwareentwicklung einordnen

Teilnehmer:innen können APIs in die Geschichte der Programmierung, Computernetzwerke, verteilten Systeme und Softwarearchitektur einordnen. Die Entwicklung von lokalen APIs zu netzwerkbasierten APIs und die aktuelle API Landschaft werden in ihrem Gesamtkontext verstanden.

LZ 1-2: Unterschiedliche Integrationsstile und -konzepte vergleichen

Unterschiedliche Ansätze zur Systemintegration sind den Teilnehmer:innen bekannt und können gegeneinander abgegrenzt werden, insbesondere:

- Integration über eine Datenbank
- Dateibasierte Systemintegration
- Integration durch synchrone Kommunikation, z. B. RPC (Remote Procedure Call)
- Integration durch asynchrone Kommunikation, z. B. Message Queues

1.3. Referenzen

[\[Fishman and McLarty 2024\]](#), [\[Hohpe and Woolf 2003\]](#)

2. Wie APIs Wert erzeugen

Dauer: 60 Min.	Übungszeit: 30 Min.
----------------	---------------------

2.1. Begriffe und Konzepte

API Economy, API-Wertschöpfung, API-Geschäftsmodelle, digitale Transformation

2.2. Lernziele

LZ 2-1: Das Konzept der "API Economy" verstehen

Das Konzept der "API Economy" wird in seiner gesamten Breite verstanden und Teilnehmer:innen können beurteilen, wie ihre Produkte und Aktivitäten dort eingeordnet werden können.

LZ 2-2: Verschiedene Arten der API-Wertschöpfung kennen

Teilnehmer:innen kennen und verstehen die verschiedenen Arten, wie APIs zur Wertschöpfung beitragen können. Die verschiedenen Arten können auf der obersten Ebene unterschieden werden:

- Private APIs, die innerhalb einer Organisation eingesetzt werden
- Partner-APIs, die mit etablierten Partnern genutzt werden
- Public APIs, die als Produkte nach außen angeboten werden

Teilnehmer:innen verstehen die verschiedenen Optionen innerhalb und die Zusammenhänge zwischen diesen Kategorien und den Wert einer übergreifenden API-Strategie.

LZ 2-3: API-Geschäftsmodelle verstehen

Teilnehmer:innen haben ein detailliertes Verständnis der möglichen Geschäftsmodelle von APIs und einiger ausgewählter Beispiele. Ausgehend von den verschiedenen Arten der API-Wertschöpfung können Teilnehmer:innen bestehende Geschäftsmodelle analysieren und mit APIs gezielt ergänzen und erweitern.

LZ 2-4: APIs und digitale Transformation verstehen

Teilnehmer:innen können APIs in das grössere Bild digitaler Transformation einordnen. APIs werden als notwendiger (aber nicht hinreichender) Teil einer digitalen Transformation verstanden. Andere notwendige Aspekte sind ebenfalls bekannt:

- Business-Ebene: neue Kanäle erschaffen, neue Möglichkeiten erkunden, größere Offenheit für Experimente und Anpassungen zeigen
- Organisationsebene: Abläufe beschleunigen, Abhängigkeiten zwischen Teams verringern und eine Plattform bereitstellen, die Teams dabei hilft, effektiver zu arbeiten

2.3. Referenzen

[\[API Business Models\]](#), [\[Goyal 2023\]](#)

3. API-Stile und -Technologien

Dauer: 60 Min.	Übungszeit: 30 Min.
----------------	---------------------

3.1. Begriffe und Konzepte

RPC, Ressource, Hypermedia, Query, Events, asynchrone Kommunikation, gRPC, HTTP-APIs, REST, GraphQL

3.2. Lernziele

LZ 3-1: Grundlegende API-Stile verstehen

Die folgenden API-Stile werden verstanden und ihre Vor- sowie Nachteile können gegeneinander abgewogen werden:

- RPC (Remote Procedure Call)
- Ressourcen-basiert
- Hypermedia
- Query
- Events und asynchrone Kommunikation

LZ 3-2: Populäre API-Technologien einordnen und den jeweiligen API-Stilen zuordnen

Teilnehmer:innen können die wichtigsten populären API-Technologien einordnen und kennen den jeweiligen implementierten API-Stil, z.B.:

- gRPC
- HTTP APIs
- REST
- GraphQL

LZ 3-3: Auswahlkriterien und Konsequenzen von API-Stilen und API-Technologien kennen

Teilnehmer:innen erwerben ein Verständnis für Kriterien, wann welche Stile und Technologien besser oder weniger gut passen, und welche Konsequenzen die jeweilige Auswahl mit sich bringt.

3.3. Referenzen

[\[Hohpe and Woolf 2003\]](#), [\[Porcello 2018\]](#), [\[Richardson et al. 2013\]](#)

4. API-Design

Dauer: 90 Min.	Übungszeit: 30 Min.
----------------	---------------------

4.1. Begriffe und Konzepte

Design von APIs, API-First, konsumentengetriebenes API-Design, Postel's Law, Vorwärts- und Rückwärtskompatibilität, Versionierung und Lebenszyklus von API-Produkten, JSON:API, HAL, API-Design-Styleguides, HTTP, HTTP-Caching

4.2. Lernziele

LZ 4-1: Bedeutung von API-Design verstehen

Teilnehmer:innen verstehen, weshalb ein gewissenhaftes Design von APIs von großer Bedeutung ist und welchen Einfluss dies auf die Nutzbarkeit, Wartung, Weiterentwicklung und Verbreitung hat.

Teilnehmer:innen verstehen, dass sich Anbieter:innen und Konsument:innen ohne zentrale Kontrolle weiterentwickeln können, und welche Herausforderungen daraus resultieren.

LZ 4-2: API-Design als "Outside-in"-Ansatz anwenden

Zu den wichtigsten Zielen von APIs zählt die effiziente Anbindung von Konsument:innen an die Schnittstelle des Betreibers. Das Design sollte auf die Anforderungen der Konsument:innen fokussiert sein und nicht entlang eventuell bestehender Systeme, Use Cases oder Datenbanken. Teilnehmer:innen kennen und verstehen die Ansätze von API-First und konsumentengetriebenem API-Design.

LZ 4-3: Offenheit und Erweiterbarkeit als wichtige Aspekte für die Weiterentwicklung von APIs interpretieren

Teilnehmer:innen kennen lose Kopplung, Postel's Law, die Bedeutung von Vorwärts- und Rückwärtskompatibilität und welchen Einfluss diese auf die Möglichkeit der Weiterentwicklung von APIs haben. Darüber hinaus erfahren Teilnehmer:innen, welche Konsequenzen dies für die Implementierung von APIs und deren Clients in streng und statisch typisierten Programmiersprachen hat.

LZ 4-4: Versionierung von APIs verstehen

Teilnehmer:innen erlangen ein Verständnis für verschiedene Aspekte der Versionierung und wie sie innerhalb des Lebenszyklus eines API-Produkts verwendet werden. Folgende Konzepte sind bekannt:

- Kompatibilität und die Unterscheidung zwischen Vorwärts- und Rückwärtskompatibilität
- Versionierungsidentifikation und Semantic Versioning als spezifisches Modell
- Verschiedene Arten der Versionierungsidentifikation für APIs (z.B. Domain-Namen, URI-Pfade, HTTP-Header, Information im Payload)

LZ 4-5: Good Practices für gelungenes API-Design kennen

Teilnehmer:innen lernen bewährte und verbreitete Ansätze für das Design von HTTP-APIs kennen und verstehen die Vorteile und Herausforderungen. Hierzu zählen u.a.:

- URL-Pfade
- korrekte Verwendung von HTTP-Methoden oder -Operationen für häufig benötigte Funktionalität wie Suchen, Filtern oder Sortieren

- Caching zur Performance-Verbesserung von APIs
- Formatvorschläge wie JSON:API oder HAL
- Problem Details for HTTP APIs (RFC 9457)
- API-Design-Styleguides

4.3. Referenzen

[Amundsen 2020], [Geewax 2021], [Higginbotham 2021], [HTTP Problem Details], [JSON], [Lauret 2019], [Zimmermann et al. 2022]

5. Beschreibung von APIs

Dauer: 90 Min.	Übungszeit: 30 Min.
----------------	---------------------

5.1. Begriffe und Konzepte

API-Beschreibung, OpenAPI, Protocol Buffers, AsyncAPI, GraphQL, gRPC, IDL

5.2. Lernziele

LZ 5-1: Wichtigkeit von API-Beschreibungen kennen

Teilnehmer:innen verstehen den Nutzen von API-Beschreibungen und kennen die Zielgruppen. Das Problem des "API Drift" ist bekannt. Sie können das Zusammenspiel von Design, Implementierung, Versionierung, Beschreibung und Versionierung der Beschreibung einordnen und wissen, wie diese Dinge im Idealfall sowie in der Realität praktiziert werden.

LZ 5-2: OpenAPI als Beschreibungssprache für HTTP-APIs einordnen

Teilnehmer:innen verstehen OpenAPI als eine auf HTTP-APIs spezialisierte Beschreibungssprache. Sie kennen die Geschichte von OpenAPI und wie sich die verschiedenen Versionen entwickelt haben. Teilnehmer:innen wissen, wie sich OpenAPI zur Codegenerierung auf der Anbieter:innen- oder Konsument:innenseite verwenden lässt, und welche Risiken diese Praxis hat. Die generellen Strukturen von JSON und YAML sind bekannt.

LZ 5-3: Den Aufbau von OpenAPI verstehen

Teilnehmer:innen verstehen die Hauptelemente einer OpenAPI-Beschreibung und wie diese für konkrete APIs verwendet werden.

- Ressourcen (Paths)
- Interaktionen (Operations)
- Repräsentationen
- Mechanismen wie Tags, Security, Components und Webhooks

LZ 5-4: Alternative Beschreibungssprachen kennen

OpenAPI ist spezialisiert auf HTTP-APIs. Für andere API-Stile können alternative Beschreibungssprachen mit ähnlichen generellen Zielen verwendet werden. Teilnehmer:innen kennen diese alternativen Beschreibungssprachen und können sie API-Stilen zuordnen.

- gRPC für RPC-orientierte APIs
- AsyncAPI für Event-orientierte APIs
- GraphQL für Query-orientierte APIs
- Das Konzept einer Interface Definition Language (IDL)

5.3. Referenzen

[\[AsyncAPI Specification\]](#), [\[GraphQL Specification\]](#), [\[gRPC Specification\]](#), [\[OpenAPI Specification\]](#), [\[Ponelat and Rosenstock 2022\]](#)

6. API-Lifecycle und API-Tooling

Dauer: 60 Min.	Übungszeit: 30 Min.
----------------	---------------------

6.1. Begriffe und Konzepte

API-Lifecycle, API-Produkt, Linting, Contract Testing, API-Gateway

6.2. Lernziele

LZ 6-1: API-Lifecycle verstehen

Teilnehmer:innen kennen verschiedene Schritte des Entwicklungszyklus eines API-Produkts und die typischen Aufgaben, die bei diesen Schritten anfallen. Die Aufteilung in Planung und Anforderungsanalyse, Design und Prototyping, Entwicklung, Testing und Qualitätsprüfung, Deployment und Veröffentlichung, Betrieb und Wartung sowie Verbesserung und Iteration sind bekannt. Ebenfalls bekannt sind die verschiedenen Lebensphasen eines APIs wie Prototyp, produktiver Betrieb, Deprecation und das Abschalten.

LZ 6-2: APIs als Produkte verwalten

APIs kommen oft in lose gekoppelten Szenarien zum Einsatz. Aus diesem Grund ist es sinnvoll, sie als Produkte zu managen. Teilnehmer:innen verstehen, was es bedeutet, ein API als Produkt zu verwalten. Dies beginnt beim Zielgruppenfokus, berücksichtigt Fragen der Nutzbarkeit, von Feedback sowie Verbesserung und behandelt auch Fragen von Deprecation und der Bereitstellung von Alternativen.

LZ 6-3: API-Lifecycle-Tooling kennen

Teilnehmer:innen kennen typische Tools für den Einsatz im API-Lifecycle zur Unterstützung von Produzenten und Konsumenten wie Linting, Testing (z. B. Contract Testing), Mocking, sowie Betrieb (API-Gateways). Teilnehmer:innen können mit einigen dieser Tools praktisch arbeiten und verstehen, sie in das Gesamtbild des API-Lifecycle-Tooling einzuordnen.

6.3. Referenzen

[\[Nwaiwu 2024\]](#)

7. API-Security

Dauer: 60 Min.	Übungszeit: 30 Min.
----------------	---------------------

7.1. Begriffe und Konzepte

Kommunikationssicherheit, TCP, HTTP, SSL/TLS, HTTPS, HTTP-Authentifizierung, OAuth, OpenID Connect

7.2. Lernziele

LZ 7-1: Grundlagen von Kommunikationssicherheit kennen

Teilnehmer:innen kennen die Grundlagen von Kommunikationssicherheit und können Technologien wie TCP, HTTP und TLS einordnen. Sie haben ein Bewusstsein für die Notwendigkeit sicherer Kommunikation – auch bei unternehmensinternen Schnittstellen.

LZ 7-2: Relevante Technologien im API-Umfeld verstehen

Teilnehmer:innen verstehen, wie HTTPS, HTTP-Authentisierung, OAuth und OpenID Connect funktionieren, inwiefern diese sich unterscheiden und wie die Technologien bei der Absicherung von APIs helfen.

LZ 7-3: OWASP API Security Top 10 kennen

Teilnehmer:innen kennen die OWASP API Security Top 10 und verstehen die darin beschriebenen häufigsten Sicherheitsprobleme von APIs.

7.3. Referenzen

[\[Ball 2022\]](#), [\[Madden 2020\]](#), [\[OWASP API Security Top 10\]](#)

8. APIs at Scale: Plattformen und Governance

Dauer: 60 Min.	Übungszeit: 0 Min.
----------------	--------------------

8.1. Begriffe und Konzepte

Internal Developer Platform (IDP), API-Plattform, Business-Plattform, API-Guidelines, API-Design-Plattform, Self-Service

8.2. Lernziele

LZ 8-1: Verschiedene Plattform-Begriffe vergleichen

Teilnehmer:innen kennen die Bereiche, in denen Plattformen verwendet werden. Sie verstehen die verschiedenen Begriffe und kennen die Unterschiede sowie das Ineinandergreifen.

- Internal Developer Platform (IDP)
- API-Plattform
- Business-Plattform

Teilnehmer:innen können zwischen dem mehr nach innen gerichteten Fokus einer IDP und dem sowohl nach innen als auch nach aussen gerichteten Fokus einer API Plattform unterscheiden.

LZ 8-2: API-Guidelines aufstellen

Teilnehmer:innen verstehen die Motivation von API-Guidelines. Sie kennen die Ziele, um innerhalb eines gewissen Bereiches eine Harmonisierung von API-Design und -Entwicklungspraktiken zu erreichen. Tools zur Unterstützung von API-Guidelines wie Linting sind bekannt und die Funktionsweise ist geläufig.

LZ 8-3: API-Guidelines by Example umsetzen

Die Teilnehmer:innen kennen Beispiele für API-Guidelines einiger Organisationen. Des Weiteren wissen sie, auf welche Art und Weise API-Guidelines entwickelt und gepflegt werden. Sie verstehen es als partizipativ sowie kontinuierlich gepflegtes Dokument von gelebten und unterstützten Praktiken in einer Organisation.

LZ 8-4: APIs als Team-Schnittstellen sehen

Teilnehmer:innen kennen Team Topologies als organisatorisches Modell für effektiv arbeitende Teams. Dabei ist der Fokus vor allem darauf gerichtet, an welchen Stellen von Team Topologies der Einsatz von APIs essenziell notwendig ist für die Umsetzung des Modells. Dies sind der Konsum von Diensten (X-a-a-S Modell) sowie das Anbieten von Diensten durch Plattform-Teams, die ihre Plattform im Self-Service bereitstellen wollen.

8.3. Referenzen

[Fishman and McLarty 2024], [Hohpe 2024], [Medjaoui et al. 2019]

Referenzen

Dieser Abschnitt enthält Quellenangaben, die ganz oder teilweise im Curriculum referenziert werden.

A

- [Amundsen 2020] Mike Amundsen, "Design and Build Great Web APIs: Robust, Reliable, and Resilient", Pragmatic, 2020
- [API Business Models] "ProgrammableWeb's 2020 Guide to API Business Models", ProgrammableWeb, May 2020. https://www.mulesoft.com/sites/default/files/cmm_files/2020_Guide_to_API_Business_Models.pdf
- [AsyncAPI Specification] AsyncAPI Specification. <https://www.asyncapi.com/docs/reference>

B

- [Ball 2022] Corey Ball, "Hacking APIs: Breaking Web Application Programming Interfaces", No Starch Press, 2022

F

- [Fishman and McLarty 2024] Stephen Fishman and Matt McLarty, "Unbundling the Enterprise: APIs, Optionality, and the Science of Happy Accidents", IT Revolution, 2024

G

- [Geewax 2021] JJ Geewax, "API Design Patterns", Manning Publications, 2021
- [Goyal 2023] Deepa Goyal, "API Analytics for Product Managers", Packt Publishing, 2023
- [GraphQL Specification] GraphQL Specification. <https://spec.graphql.org/>
- [gRPC Specification] gRPC Specification. <https://grpc.io/docs/>

H

- [Higginbotham 2021] James Higginbotham, "Principles of Web API Design: Delivering Value with APIs and Microservices", Addison-Wesley, 2021
- [Hohpe 2024] Gregor Hohpe, "Platform Strategy: Innovation Through Harmonization", 2024
- [Hohpe and Woolf 2003] Gregor Hohpe and Bobby Woolf, "Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions", Addison-Wesley, October 2003
- [HTTP Problem Details] Mark Nottingham, Erik Wilde, and Sanjay Dalal, "Problem Details for HTTP APIs", Internet Proposed Standard RFC 9457, July 2023

J

- [JSON] Tim Bray, "The JavaScript Object Notation (JSON) Data Interchange Format", Internet Standard RFC 8259, December 2017

L

- [Lauret 2019] Arnaud Lauret, "The Design of Web APIs", Manning Publications, 2019

M

- [Medjaoui et al. 2019] Mehdi Medjaoui, Erik Wilde, Ronnie Mitra, and Mike Amundsen, "Continuous API Management", O'Reilly, 2nd Edition, October 2021
- [Madden 2020] Neil Madden, "API Security in Action", Manning Publications, 2020

N

- [Nwaiwu 2024] Ikenna Nwaiwu, "Automating API Delivery: APIOps with OpenAPI", Manning Publications, July 2024

O

- [OpenAPI Specification] OpenAPI Specification. <https://spec.openapis.org/oas/latest.html>
- [OWASP API Security Top 10] "OWASP API Security Project", Open Worldwide Application Security Project. <https://owasp.org/www-project-api-security/>

P

- [Ponelat and Rosenstock 2022] Joshua S. Ponelat and Lukas L. Rosenstock, "Designing APIs with Swagger and OpenAPI", Manning Publications, 2022
- [Porcello 2018] Eve Porcello, "Learning GraphQL: Declarative Data Fetching for Modern Web Apps", O'Reilly, 2018

R

- [Richardson et al. 2013] Leonard Richardson, "RESTful Web APIs: Services for a Changing World", O'Reilly, 2013

Z

- [Zimmermann et al. 2022] Olaf Zimmermann, Mirko Stocker, Daniel Lübke, Uwe Zdun, Cesare Pautasso, "Patterns for API Design: Simplifying Integration with Loosely Coupled Message Exchanges", Addison-Wesley Professional, November 2022, ISBN 978-0-13767-010-9