Qu'est-ce qu'une API ?

API, l'acronyme anglais de « *Application Programming Interface* », signifie « interface de programmation d'application » en français. Les API sont un ensemble de fonctions et de procédures qui permettent de créer des applications. Elles accèdent aux données et aux fonctionnalités d'autres applications, services ou systèmes d'exploitation.

En fait, elles servent d'intermédiaire entre les plateformes informatiques. Elles permettent à deux applications sans rapport de « dialoguer » entre elles en s'échangeant des données ou des services. Dans le contexte des API, le terme « application » désigne un logiciel avec une fonction distincte.

Les API sont constituées d'un ensemble de définitions et de protocoles favorisant la création et l'intégration de logiciels d'applications. Ces fonctions facilitent l'accès aux services d'une application grâce à un langage de programmation qui permet d'effectuer des requêtes. Les API sont implémentées par des appels de fonction. Il s'agit d'instructions de langage qui demandent à un logiciel de réaliser des actions et des services particuliers.

Plus précisément, on parle d'API lorsqu'une entité informatique cherche à interagir avec un système, et que cette interaction suit des normes et respecte les contraintes d'accès définies par le système tiers.

Comportant un ensemble de classes, de méthodes et de constantes, les API représentent une façade grâce à laquelle une application propose des services à d'autres logiciels. Elles sont fournies par une bibliothèque logicielle ou un service Web.

Dans le contexte actuel, les API jouent un rôle crucial dans la connexion entre les intelligences artificielles et les environnements de travail. Elles permettent aux

solutions d'IA d'accéder aux données d'entreprise, d'automatiser des tâches répétitives et d'enrichir les processus métier avec des capacités d'analyse prédictive et de traitement du langage naturel.

Par ailleurs, les API rendent les développeurs de logiciels plus productifs. Sans elles, les développeurs devraient écrire et maintenir leur propre code pour accéder aux ressources externes. Le fait de disposer d'une méthode standard pour lire et écrire dans ces ressources rend la plateforme qui fournit l'API plus accessible et plus attrayante pour les développeurs, ce qui augmente la probabilité que des tiers utilisent et échangent des données avec leur plateforme.

Il est également possible de classer les API selon les systèmes pour lesquels elles sont conçues. Il existe différentes catégories par cas d'utilisation :

- les API de base de données qui permettent à une application de communiquer avec un système de gestion de base de données ;
- les API Web qui correspondent aux interfaces de programmation pour un serveur web ou un navigateur web ;
- les API des systèmes d'exploitation qui définissent la façon dont les applications doivent utiliser les données et les services d'un système d'exploitation ;
- les API distantes qui sont créées pour interagir via un réseau de communication.

Comment fonctionne une API ?

Les API permettent d'accéder aux fonctions ou aux données d'une application à distance à partir d'une autre application, par l'intermédiaire d'une interface applicative standard. Le logiciel sollicité reçoit une requête dans un langage

universel. Ce langage employé par les API permet au logiciel cible de comprendre la requête, puis de transmettre les données demandées.

Les API permettent de faire communiquer un produit ou service avec d'autres sans connaître les détails de leur fonctionnement. Elles facilitent le développement d'applications, ce qui fait gagner du temps et de l'argent et réduit le temps d'intégration. Utilisées dans de nombreux programmes, elles ressemblent à un jeu de construction et offrent des pièces de fonctionnalités pouvant être intégrées dans diverses applications.

Par exemple, lorsqu'une personne se connecte à Facebook depuis son téléphone, l'application envoie un appel à une API, ou API call, afin de récupérer les informations liées à son compte et à son identification. Les serveurs de Facebook fournissent alors ces données et les renvoient à l'application mobile.

Ce type d'API, appelée API Web, est la plus courante, mais elle est limitée au web.

De plus, les API reposent sur une architecture client-serveur. Ainsi, le client désigne l'application qui adresse la requête, tandis que le serveur correspond à l'application qui envoie la réponse. Les API fonctionnent sur différents styles d'architecture dont les plus connus sont REST (Representational State Transfer), RPC (Remote Procedure Calls) et SOAP (Simple Object Access Protocol).

Les API REST permettent au client d'effectuer une demande sous forme de données à un serveur. En utilisant les données du client pour réaliser des fonctions internes, le serveur transmet les données de sortie. Les API REST peuvent s'adapter à différents formats de données comme JSON ou XLM.

Les API RPC sont la forme la plus ancienne et la plus simple d'interaction API. Elles sont appelées appels de procédure à distance. Le client exécute une fonction ou une procédure sur le serveur, puis le serveur lui renvoie la sortie. On retrouve les styles d'architecture RPC dans des technologies API comme GraphQL et gRPC.

Les API SOAP sont les API qui utilisent le protocole simple d'accès aux objets. Leur modèle d'architecture repose sur une communication entre le client et le serveur via XML.

Il existe aussi des styles d'architecture orientés événements, comme le webhook, qui sont également appelés architectures événementielles ou de streaming. Ce sont des modèles qui visent à répondre à un événement en temps réel.

Quels sont les différents types d'API ?

- Les API publiques
- Les API privées
- Les API partenaires
- Les API composites

Les API publiques

Également appelées API externes ou publiques, les API ouvertes présentent des mesures de sécurité assouplies, permettant aux développeurs et aux utilisateurs externes d'accéder facilement aux données d'une entreprise.

En effet, ces API sont à la portée de tous les tiers et peuvent être utilisées sans restriction. Ce type d'API peut être employé librement par les développeurs tiers pour créer puis tester des applications, mais aussi leur donne les moyens d'innover.

Les API publiques peuvent aussi être commerciales. Dans ce cas, des frais d'abonnement sont facturés ou utilisés sur une base de paiement à l'utilisation. Les essais gratuits offerts par les éditeurs permettent aux utilisateurs d'évaluer les API avant de s'abonner.

L'API de Google Maps est un bon exemple d'API publique. Celle-ci permet aux utilisateurs de bénéficier de fonctionnalités de suivi et de géolocalisation. Ces services sont utilisés dans des applications de covoiturage ou de livraison de repas par exemple.

Les API privées

Également connues sous le nom d'API internes, les API privées sont cachées aux parties externes et utilisées pour améliorer la communication au sein d'une entreprise. Grâce à cette méthode, les entreprises peuvent simplifier le partage des données entre les différents services et tous les sites commerciaux.

Bien que l'accès soit limité aux opérations internes, les API privées prévoient toujours des mesures de sécurité pour vérifier l'identité des employés avant d'autoriser l'accès au système.

Par exemple, elles peuvent connecter les systèmes de paie et RH. Grâce à cette approche, les organisations peuvent contrôler totalement l'utilisation d'une API. Les développeurs internes peuvent aussi recourir à ce type d'API pour développer des applications destinées aux clients.

Les API privées facilitent les actions qui touchent les entreprises dans leur ensemble. En utilisation et en réutilisant ces API, les entreprises développent leur productivité et leur efficacité. Par exemple, une équipe d'un centre d'appel ayant créé une API d'informations client permettant d'accéder à leur nom et à leurs coordonnées peut la réutiliser dans une application orientée client.

Les API partenaires

Situées à mi-chemin entre les API publiques et privées, les API partenaires désignent les API qui sont partagées avec certains partenaires commerciaux ou stratégiques d'une entreprise. Elles sont accessibles par l'intermédiaire d'une licence ou de droits d'accès spécifiques.

En effet, les API partenaires sont plus limitées quant aux personnes qui peuvent accéder au service. Elles peuvent être gratuites ou payantes. Étant donné que les API partenaires ne sont mises à la disposition que de certaines parties, elles ont tendance à avoir des règles plus strictes et plus rigoureuses en matière d'autorisation, d'authentification et de sécurité.

Certaines des API les plus importantes et les plus utilisées sont des API partenaires, l'API d'eBay en est un exemple.

Autoriser l'accès aux données à ses partenaires permet à l'entreprise de surveiller la manière dont ils utilisent les actifs numériques. Le but est notamment de s'assurer que les API employées augmentent l'expérience utilisateur. Les API partenaires sont configurées de manière à ce que chaque organisation puisse avoir accès aux données. Cette catégorie d'API permet de créer de nouveaux flux de revenus sans compromettre la sécurité.

Les API composites

Ce type d'API combine deux ou plusieurs interfaces de programmation de données et de services afin de créer une séquence d'opérations connexes ou interdépendantes. Les API sont donc regroupées au sein d'un seul appel d'API. Ainsi, un seul appel est effectué vers le serveur au lieu de plusieurs, et une seule réponse est transmise. Les API composites peuvent être utilisées pour traiter les exigences ou les comportements complexes d'un système. Par rapport aux API individuelles, elles peuvent améliorer la vitesse et les performances d'un logiciel.

Les API composites sont particulièrement utiles dans les architectures de microservices, où un utilisateur peut avoir besoin d'informations provenant de plusieurs services pour effectuer une seule tâche.

Le guide pour comprendre les langages HTML et CSS.

Découvrez comment utiliser les langages HTML et CSS pour gérer votre site web efficacement.

- Balises HTML à connaître
- Mettre à jour un fichier CSS
- Les écueils à éviter
- Ressources pédagogiques

Télécharger

En savoir plus

Comment utiliser une API ?

Si l'utilisation d'une API peut sembler un effort supplémentaire inutile, elle a pour but d'améliorer la sécurité des informations et de faciliter l'accès aux données requises pour prendre des décisions économiques judicieuses.

Malheureusement, la facilité d'utilisation est relative. Les personnes qui ne sont pas des professionnels de l'informatique peuvent être découragées par la perspective d'apprendre à utiliser une API.

Choisir une API appropriée

Avant d'utiliser une API, les développeurs doivent s'assurer d'en sélectionner une qui est appropriée. Une API appropriée est une API qui peut rapporter de meilleurs bénéfices à une entreprise. Le choix de l'API dépend des éléments suivants :

- Le modèle de programmation utilisé ;
- L'objectif de l'API ;
- Les types de données à échanger ;
- La structure de communication ;
- Le langage de programmation dans lequel les données et les services sont écrits ;
- Le niveau de sécurité requis ;
- Le budget.

De plus, de nombreuses API publiques sont disponibles avec un modèle freemium. Pour les personnes qui débutent dans l'utilisation d'une API, il est intéressant de commencer par ce type d'interface afin de se familiariser avec l'univers de la programmation.

Consulter la documentation de l'API

Examiner la documentation est absolument indispensable pour utiliser efficacement l'API choisie. Un manuel de référence fournit tout ce que les développeurs doivent connaître sur les fonctionnalités de l'interface de programmation. La documentation contient les éléments suivants :

- Un guide de démarrage rapide ;
- Les informations d'authentification comme l'obtention et l'utilisation d'une clé API ;
- Des explications sur les appels et l'envoi de requêtes à l'aide de l'API ;
- La liste des ressources fournies par les serveurs.

La documentation de l'API est en général divisée en deux colonnes : humain et machine. Sur la colonne humaine, on trouve des descriptions d'API. Quant à la colonne de la machine, elle dispose d'une console pour passer les appels et comporte des informations qui intéressent les clients et les serveurs lors du test API. Pour aider les développeurs à apprendre à utiliser une interface de programmation, la documentation fournit également des exemples et des tutoriels.

En résumé, une bonne documentation est très utile pour les développeurs qui souhaitent savoir faire bon usage d'une API.

Authentifier l'API

Pour de nombreuses API, une authentification est nécessaire. Il s'agit d'un compte utilisateur qui permet d'accéder aux données. L'authentification permet aux API de contrôler l'accès aux ressources. Les moyens d'authentification les plus courants sont le nom d'utilisateur et le mot de passe.

Pour pouvoir se connecter à une interface de programmation, il faut aussi une clé API. C'est une étape primordiale pour confirmer son identité. Cet identifiant unique joue un rôle important lors des appels API. Il est utilisé dans une requête pour authentifier un utilisateur, un développeur ou un programme. La clé API détermine les ressources auxquelles l'utilisateur ou le développeur est autorisé à accéder depuis le serveur. Pour éviter les comportements frauduleux, il est important de sécuriser la clé API avec un mot de passe fort. En cas de violation, il est possible de se procurer une nouvelle clé.

Pour récupérer des données, il faut envoyer une combinaison unique de caractères et de chiffres avec chaque demande au serveur. De ce fait, après avoir pris connaissance de l'API utilisée, les utilisateurs et les développeurs doivent accéder aux détails d'authentification. Pour que le fournisseur de services puisse leur fournir ces détails, ils doivent créer un compte auprès de lui ou vérifier leur identité.

Préparer les informations de demande

Maintenant que les utilisateurs savent comment accéder aux données à l'aide de l'API, le moment est venu de préparer une demande. Celle-ci doit comporter trois parties :

- le type de demande ou d'action qui permet de déterminer ce que les utilisateurs veulent faire avec les données en utilisant des méthodes de requête HTTP ;
- l'URL ou point de terminaison qui fournit l'emplacement numérique d'une ressource à partir duquel l'API reçoit des requêtes et envoie des réponses ;
- les paramètres qui correspondent aux contraintes ou exigences supplémentaires nécessaires à l'API pour répondre à une demande.

Pour préparer une demande, le moyen le plus facile et le plus rapide est d'utiliser un client REST. Une fois la demande prête, il ne reste plus qu'à l'envoyer via son client REST, son navigateur Web ou sa ligne de commande. Les informations seront transmises à la structure présentée dans la documentation de l'API.

Faire une requête auprès de l'API

Après avoir préparé la demande, les utilisateurs vont pouvoir faire une enquête API. Pour cela, la manière la plus pratique est d'utiliser un client HTTP. C'est un excellent moyen pour structurer et transmettre la demande. Pour consulter une API, il suffit de lui adresser une requête à partir d'un logiciel adapté. Pour effectuer un appel d'API, il faut prendre en compte un certain nombre de paramètres comme l'URL, ou point de terminaison. Une API propose des services ayant chacun leur propre URL. Le point de terminaison de l'API permet d'effectuer un appel qui transmet la requête et les informations demandées au serveur. Pour préciser la requête, différents paramètres viennent compléter l'URL. Une fois la requête soumise, les utilisateurs recevront de l'API une réponse structurée avec les informations demandées.

Synchroniser une application avec l'API

Une fois l'API parfaitement assimilée, il est temps pour les développeurs de connecter leur application avec l'interface de programmation d'application choisie. Pour cela, ils doivent connaître les langages de programmation comme Java, JavaScript, Python ou encore PHP. De plus, l'intégration de l'API doit se faire en douceur. Si les développeurs utilisent une API spécifique telle que l'API REST, celle-ci sera facile à pratiquer par rapport aux autres interfaces de programmation.

Comment sécuriser une API ?

La sécurisation des API constitue un enjeu majeur pour les entreprises, car ces interfaces représentent des points d'entrée privilégiés pour les cyberattaques. Une API mal sécurisée peut exposer des données sensibles, compromettre l'intégrité du système et causer des perturbations majeures dans les opérations commerciales.

Quelles sont les bonnes pratiques pour sécuriser une API ?

La mise en place d'une stratégie de sécurité robuste pour les API nécessite l'adoption de plusieurs pratiques complémentaires qui forment un écosystème de protection multicouche.

Rotation automatique des clés API

La rotation automatique des clés API consiste à renouveler périodiquement les identifiants d'accès pour limiter la fenêtre d'exposition en cas de compromission. Cette pratique réduit considérablement les risques liés à l'utilisation prolongée d'une même clé. Les organisations implémentent généralement des cycles de rotation de 30 à 90 jours, selon le niveau de sensibilité des données. Les systèmes automatisés facilitent cette rotation en générant de nouvelles clés, en notifiant les utilisateurs autorisés et en révoquant les anciennes clés après une période de grâce.

Chiffrement bout-en-bout

Le chiffrement bout-en-bout protège les données pendant leur transmission entre le client et le serveur. L'utilisation du protocole HTTPS avec des certificats SSL/TLS récents garantit que les informations échangées restent confidentielles et intègres. Cette protection s'étend également au stockage des données sensibles avec des algorithmes de chiffrement robustes comme AES-256. Les entreprises doivent s'assurer que les clés de chiffrement sont stockées séparément des données et font l'objet d'une gestion rigoureuse.

Monitoring et détection d'intrusions

Le monitoring en temps réel des API permet de détecter rapidement les comportements anormaux et les tentatives d'intrusion. Cette surveillance est essentielle car plus une intrusion est détectée tôt, moins les dommages potentiels sont importants. Les systèmes de détection analysent les patterns de trafic, identifient les anomalies dans les requêtes et alertent les équipes de sécurité en cas de comportement suspect. Les métriques surveillées incluent le volume de requêtes, les codes de réponse d'erreur, les tentatives d'accès non autorisées et les latences inhabituelles.

Quelles sont les menaces dont il faut particulièrement se méfier avec une API ?

Les API font face à des menaces spécifiques qui nécessitent une vigilance particulière et des mesures de protection adaptées.

Injection de code malveillant

L'injection de code malveillant représente une menace majeure où les attaquants exploitent les vulnérabilités de validation des données d'entrée pour exécuter du code non autorisé. Les injections SQL, NoSQL ou de commandes système peuvent compromettre entièrement la base de données ou le serveur. La prévention passe par la validation stricte des données d'entrée, l'utilisation de requêtes préparées et la mise en place de listes blanches pour les paramètres autorisés. Les développeurs doivent également implémenter des mécanismes d'échappement appropriés pour neutraliser les caractères spéciaux.

DDoS

Les attaques par déni de service distribué (DDoS) visent à rendre une API indisponible en la saturant de requêtes simultanées. Ces attaques peuvent paralyser complètement les services et causer des pertes économiques importantes. La protection contre les DDoS implique la mise en place de systèmes de limitation du taux de requêtes (rate limiting), l'utilisation de réseaux de distribution de contenu (CDN) et l'implémentation de mécanismes de détection automatique des patterns d'attaque. Les solutions cloud offrent souvent des protections DDoS intégrées avec une capacité d'absorption élevée.

Vol de tokens

Le vol de tokens d'authentification permet aux attaquants d'usurper l'identité d'utilisateurs légitimes et d'accéder aux ressources protégées. Cette menace est particulièrement critique car elle peut passer inaperçue pendant de longues périodes. La protection contre le vol de tokens inclut l'utilisation de tokens à durée de vie limitée, l'implémentation de mécanismes de révocation immédiate, le stockage sécurisé côté client et la surveillance des patterns d'utilisation anormaux. Les tokens

JWT (JSON Web Tokens) doivent être correctement signés et les informations sensibles ne doivent jamais être incluses dans le payload.