WebAPI

01 - Préambule : Le http

TBDC
Thomas BUREAU DU COLOMBIER

Structure digitale

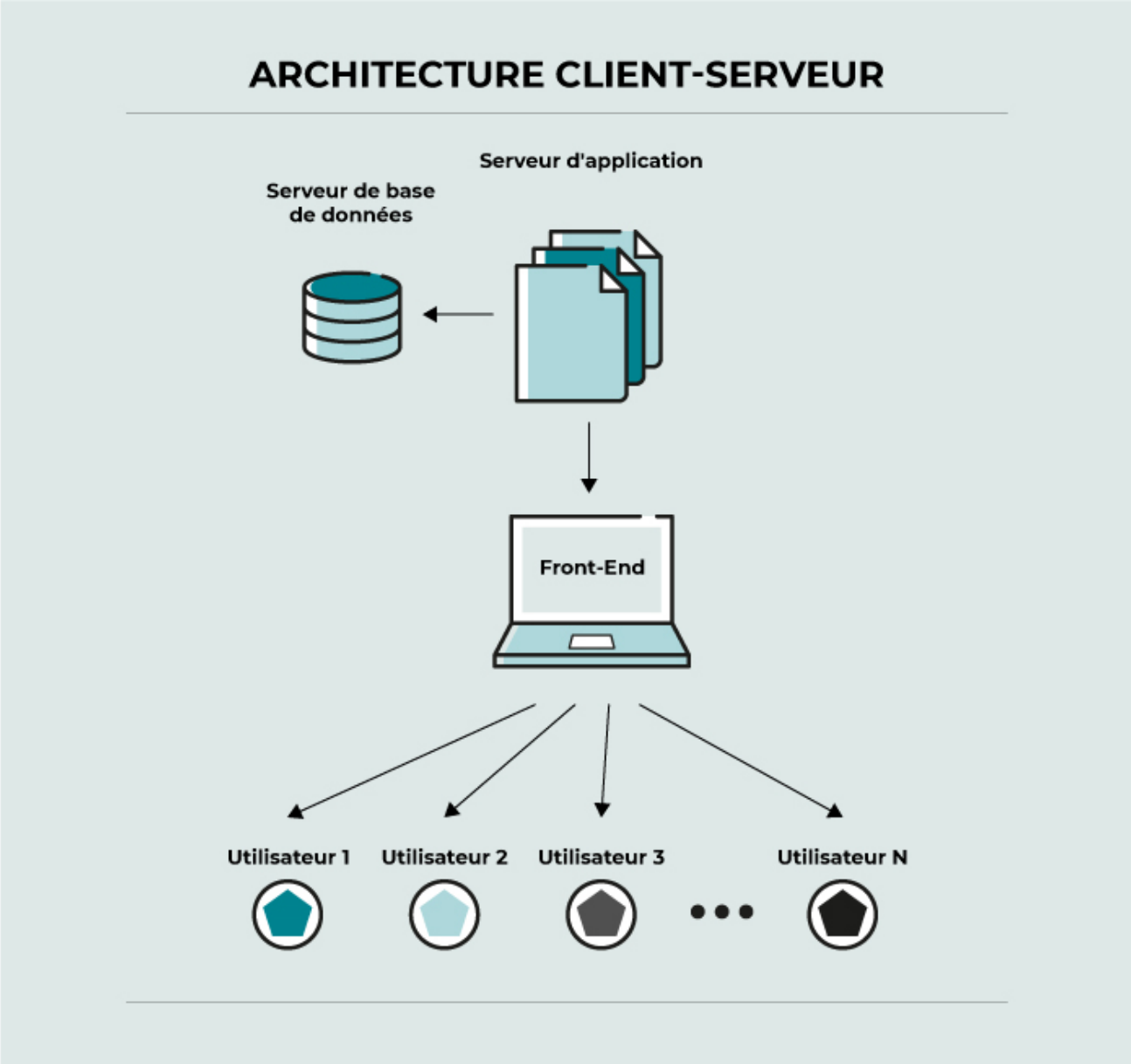
Client, serveur,
La base...

TBDC
Thomas BUREAU DU COLOMBIER

Discutons ...

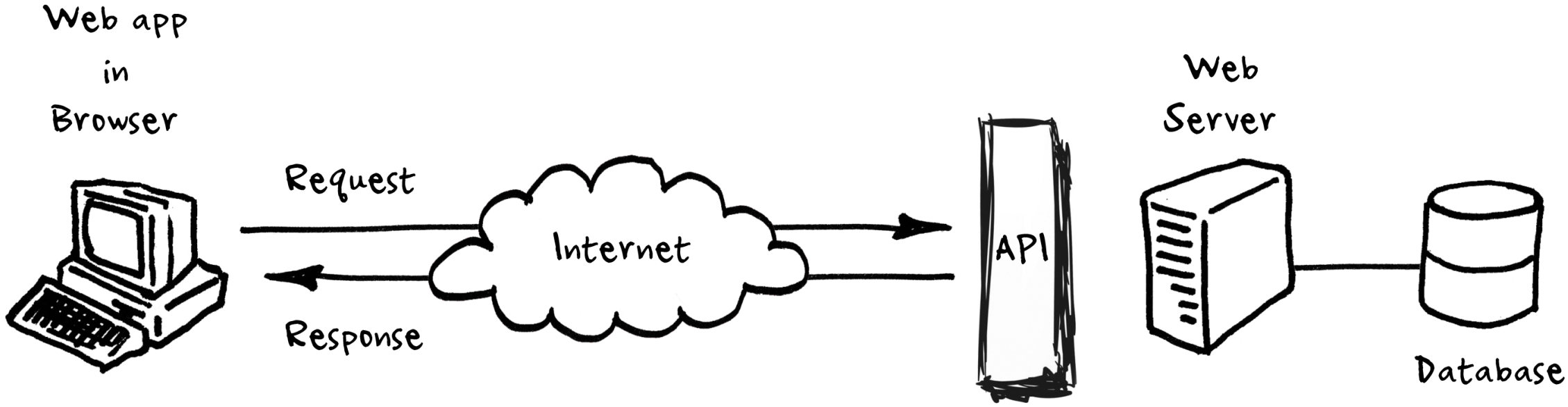
Quels sont les appareils de vos clients ?
Et votre configuration ?

Client - serveur



TBDC
Thomas BUREAU DU COLOMBIER

Avec une APi


Web app
in
Browser
Request
Response
Internet
API
Web
Server
Database

Echanger

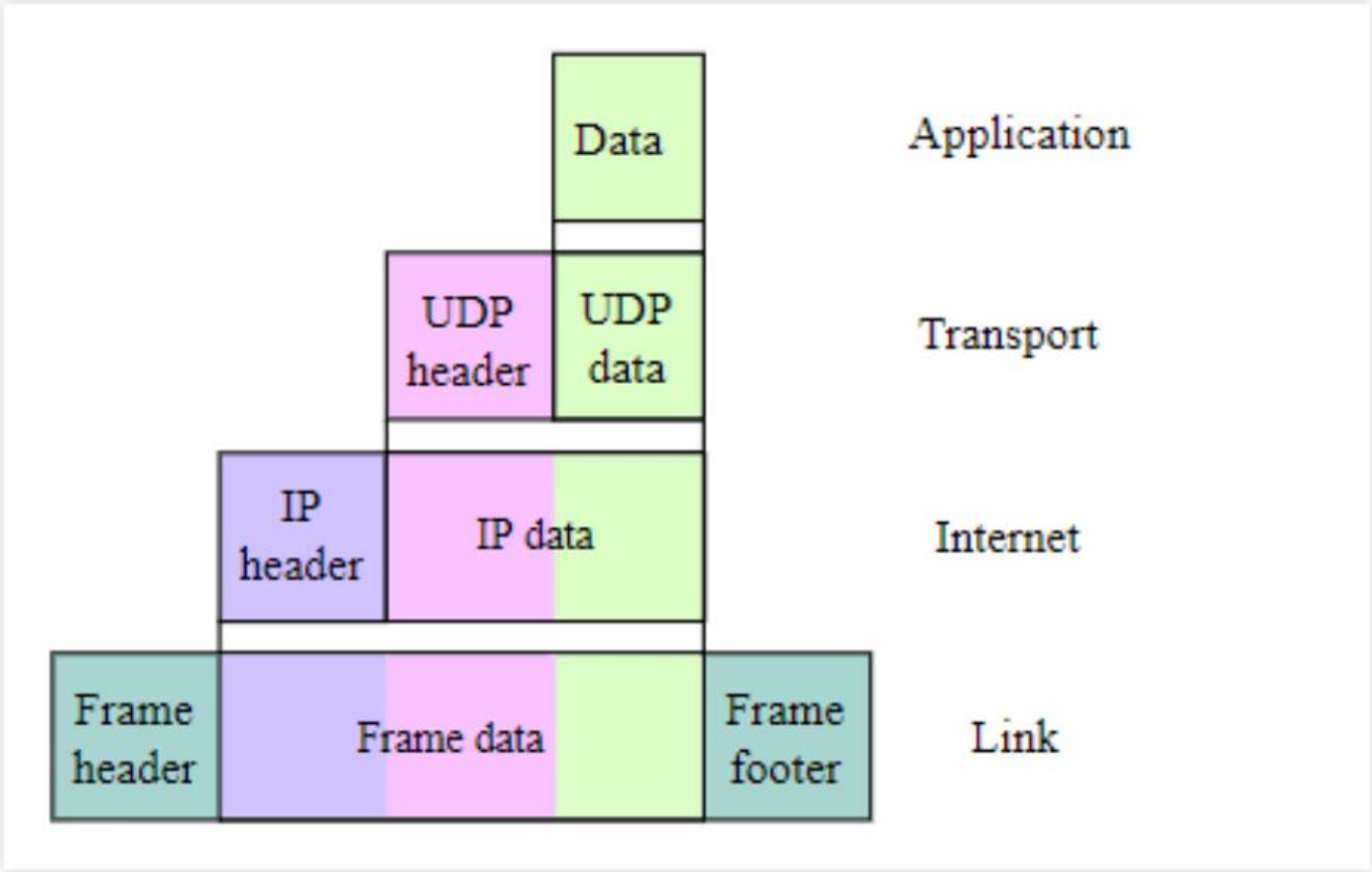
Comment on échange nos données

TBDC
Thomas BUREAU DU COLOMBIER

Modèle OSI et TCP/IP

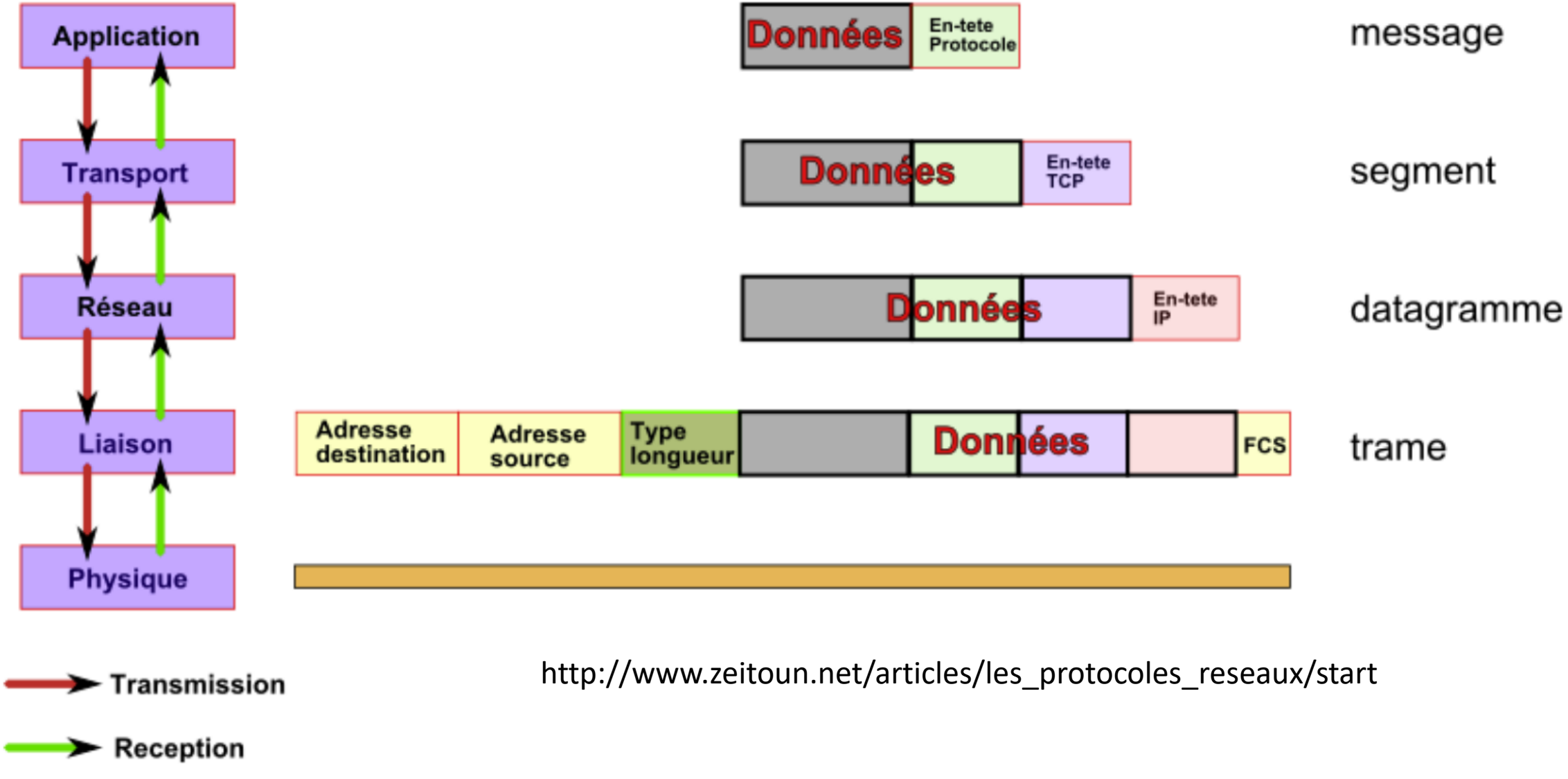
Couche	Modèle OSI	Modèle TCP/IP	Description	Exemples de protocoles du modèle TCP/IP
7	couche application	Couche application	Elle assure l'interface avec les applications. Il s'agit donc du niveau le plus proche des utilisateurs, géré directement par les logiciels.	FTP, SSH, SFTP, DNS,HTTP, H323, IMAP, NFS,POP3, Samba, SNMP, JXTA, RIP,SMTP, Telnet, FIX
6	couche présentation			
5	couche session			
4	couche de transport	Couche Transport (TCP)	Elle est chargée du transport des données, de leur découpage en paquets et de la gestion des éventuelles erreurs de transmission. Les protocoles de transport déterminent aussi à quelle application chaque paquet de données doit être délivré.	TCP, UDP, RTP
3	couche de réseau	Couche Internet (IP)	Elle permet de gérer l'adressage et le routage des données, c'est-à-dire leur acheminement via le réseau. Elle permet l'acheminement des datagrammes (paquets de données) vers des machines distantes ainsi que de la gestion de leur fragmentation et de leur assemblage à réception.	IP, ICMP, IGMP, ARP
2	couche de liaison des données	Couche accès réseau ou liaison	Elle spécifie comment les paquets sont transportés sur la couche physique, et en particulier les séquences particulières de bits qui marquent le début et la fin des paquets (le tramage). Cette couche est parfois subdivisée en deux sous couches nommées LLC et MAC.	Ethernet, ATM, Token ring, SLIP
1	couche physique	Couche physique	Elle décrit les caractéristiques physiques de la communication comme les conventions à propos de la nature du medium utilisé pour les communications (les câbles, les liens par fibre optique ou par radio), et tous les détails associés comme les connecteurs, les types de codage ou de modulation, le niveau des signaux, les longueurs d'ondes, la synchronisation et les distances maximales.	électronique, radio, laser

Encapsulation des données



Encapsultaion progressive des données échangées

Modèle TCP/IP

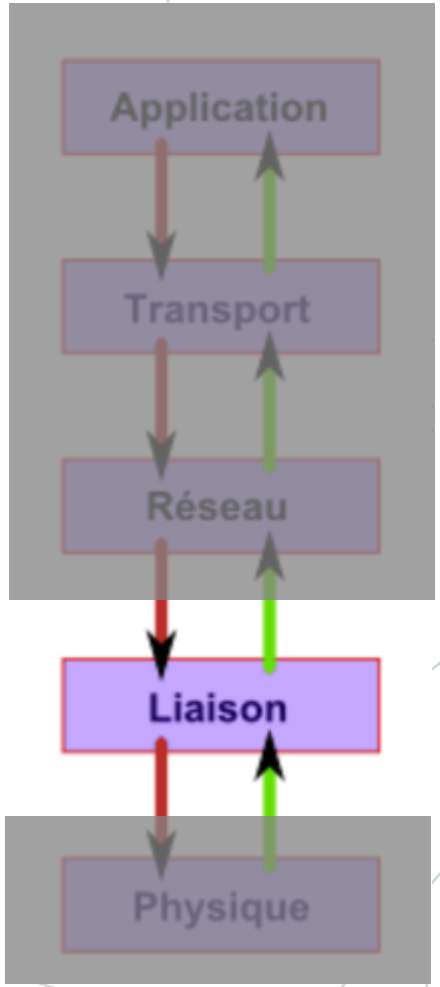


http://www.zeitoun.net/articles/les_protocoles_reseaux/start

Exemple d’une requête http vers un site

- Mise en place d’un sniffeur sur une requête HTTP
→"Ethereal"

Exemple d'une requête http vers un site



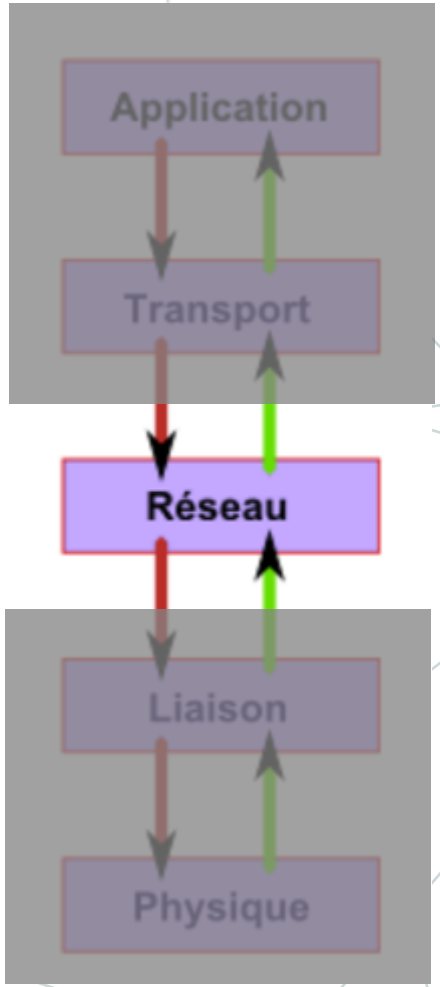
Au niveau de la couche "accès réseau"

Nous observons le protocole Ethernet dont les deux attributs importants sont les adresses physiques (MAC) des cartes réseaux.

```
Ethernet II, Src: AsustekC_c9:f7:78 (00:11:d8:c9:f7:78), Dst: 3com_44:8c:88 (00:0a:5e:44:8c:88)
    Destination: 3com_44:8c:88 (00:0a:5e:44:8c:88)
    Source: AsustekC_c9:f7:78 (00:11:d8:c9:f7:78)
    Type: IP (0x0800)
```

TBDC
Thomas BUREAU DU COLOMBIER

Exemple d'une requête http vers un site



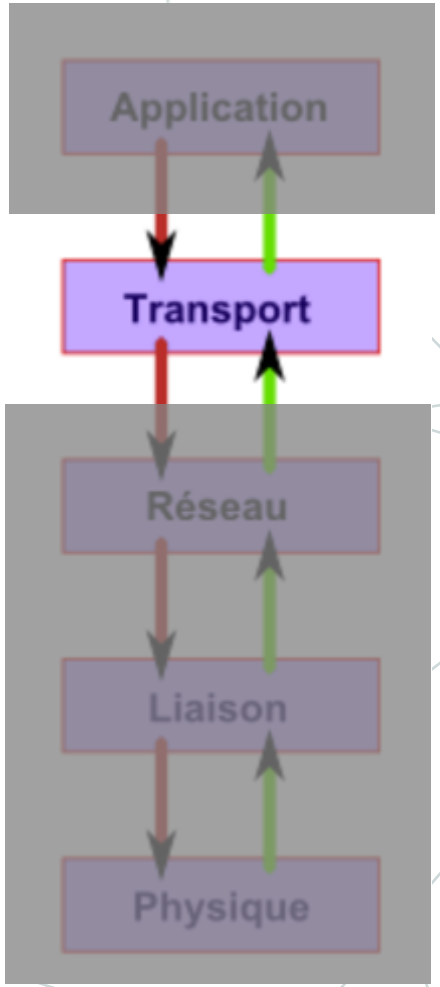
Au niveau de la couche "Internet"

Nous observons le protocole IP dont les deux attributs importants sont les adresses IP source et destination.

```
Internet Protocol, Src: 192.168.0.70 (192.168.0.70), Dst: 213.251.133.196 (213.251.133.196)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 500
    Identification: 0x4152 (16722)
    Flags: 0x04 (Don't Fragment)
        0... = Reserved bit: Not set
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (0x06)
    Header checksum: 0xdb03 [correct]
    Source: 192.168.0.70 (192.168.0.70)
    Destination: 213.251.133.196 (213.251.133.196)
```

TBDC
Thomas BUREAU DU COLOMBIER

Exemple d'une requête http vers un site

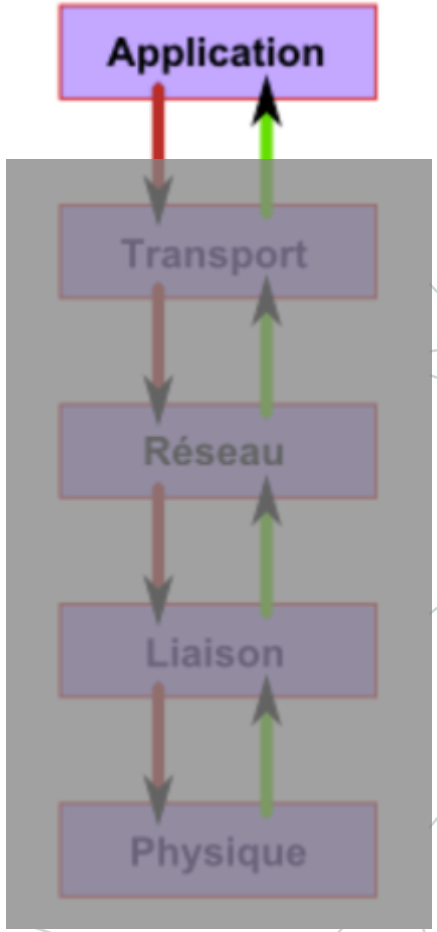


Au niveau de la couche "transport"

Nous observons le protocole TCP qui contient les informations des ports de la connexion.

```
Transmission Control Protocol, Src Port: 33276 (33276), Dst Port: www (80), Seq: 1, Ack: 1, Len: 448
    Source port: 33276 (33276)
    Destination port: www (80)
    Sequence number: 1    (relative sequence number)
    Next sequence number: 449    (relative sequence number)
    Acknowledgement number: 1    (relative ack number)
    Header length: 32 bytes
    Flags: 0x0018 (PSH, ACK)
        0... .... = Congestion Window Reduced (CWR): Not set
        .0.. .... = ECN-Echo: Not set
        ..0. .... = Urgent: Not set
        ...1 .... = Acknowledgment: Set
        .... 1... = Push: Set
        .... .0.. = Reset: Not set
        .... ..0. = Syn: Not set
        .... ...0 = Fin: Not set
    Window size: 5840 (scaled)
    Checksum: 0x1e95 [incorrect, should be 0x989d]
    Options: (12 bytes)
        NOP
        NOP
        Time stamp: tsval 36207655, tsecr 148502841
```

Exemple d'une requête http vers un site



Au niveau de la couche "application"

Nous observons le protocole HTTP : un client web tente de se connecter sur le site web www.zeitoun.net en utilisant la méthode GET.

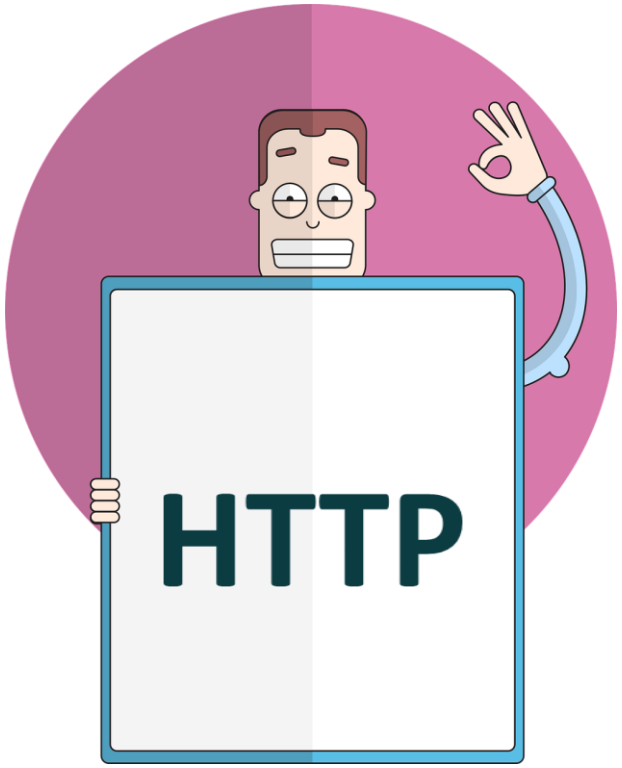
```
Hypertext Transfer Protocol
    GET / HTTP/1.1\r\n
        Request Method: GET
        Request URI: /
        Request Version: HTTP/1.1
    Host: www.zeitoun.net\r\n
    User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.7.10) Gecko/20050716 Firefox/1.0.6\r\n
    Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5\r\n
    Accept-Language: en-us,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    Cookie: PHPSESSID=cd212b11276ada1bfa58dce3090575c2\r\n
    \r\n
```

TBDC
Thomas BUREAU DU COLOMBIER

KESAKO HTTP

TBDC
Thomas BUREAU DU COLOMBIER

Le protocole HTTP



Définition

Protocole (informatique) :

Un protocole informatique est un ensemble de règles (ou normes) créées pour permettre aux ordinateurs de s'interconnecter et aux périphériques d'échanger des données.

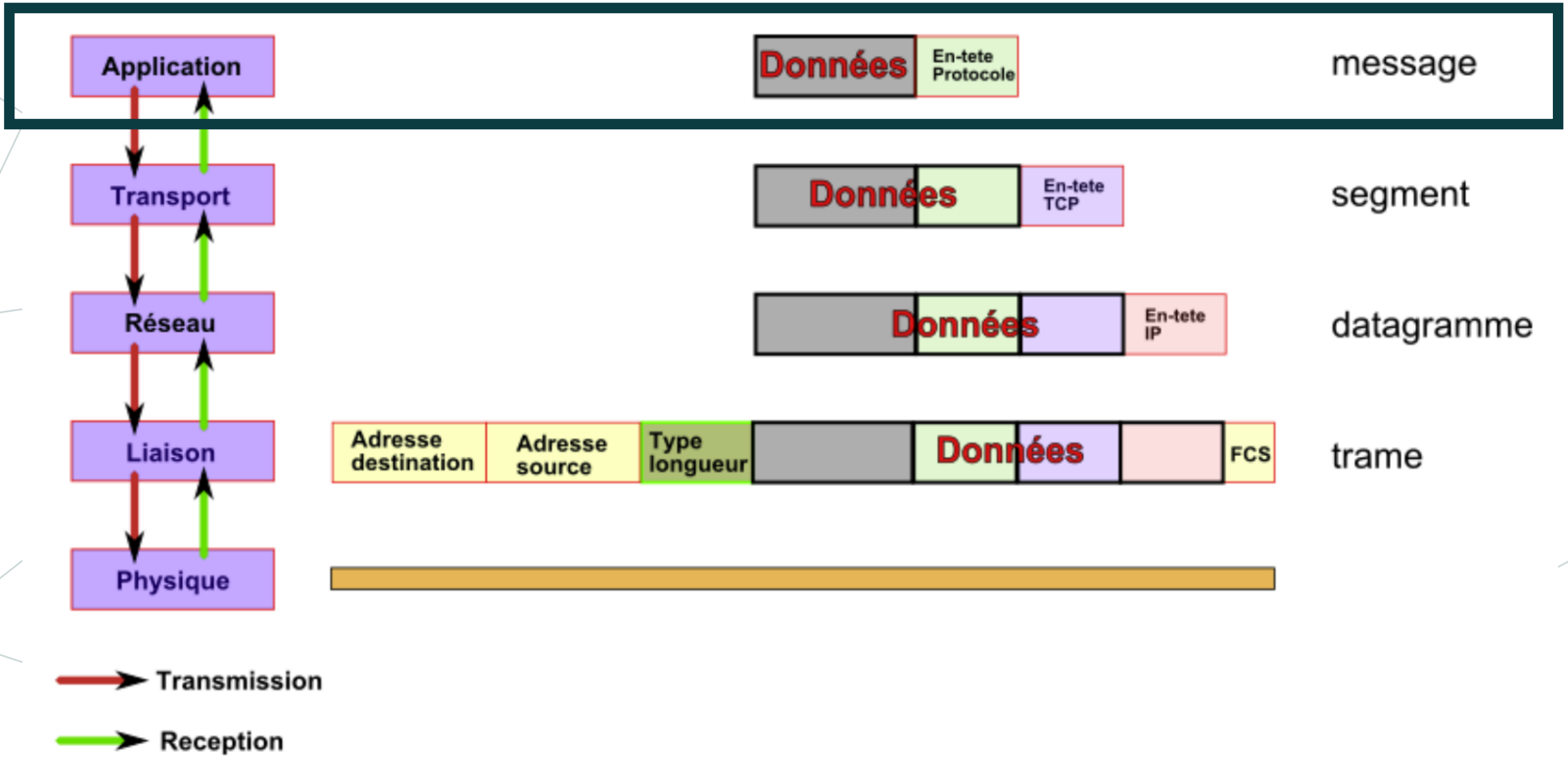
Le **protocole HTTP** (*Hypertext Transfer Protocol*) s'inscrit dans la suite TCP/IP et se situe dans la couche application du modèle OSI.

Ce protocole a été conçu et développé par Tim Berners-Lee à partir de 1989. La première version documentée de HTTP est apparue en 1991.

Source : https://www.schoolmouv.fr/enseignants/cours/la-page-web-http-et-langages-html-et-css-/fiche-de-cours

Où on est ?

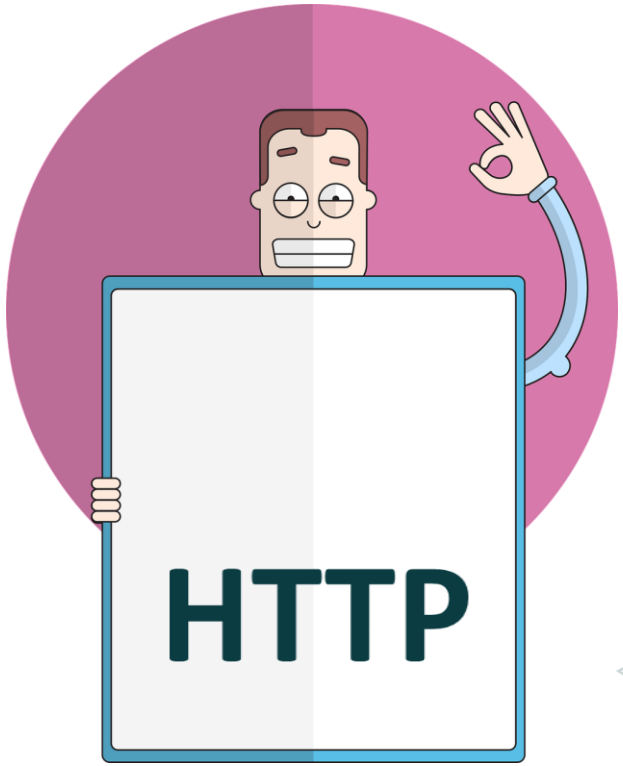
On est ici !!



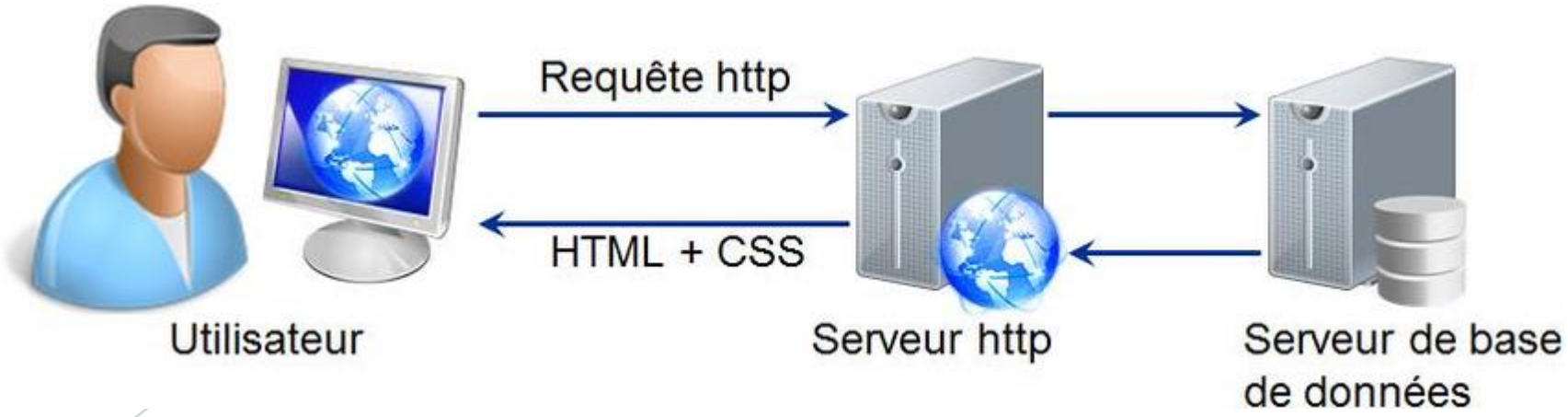
Le protocole HTTP

À retenir

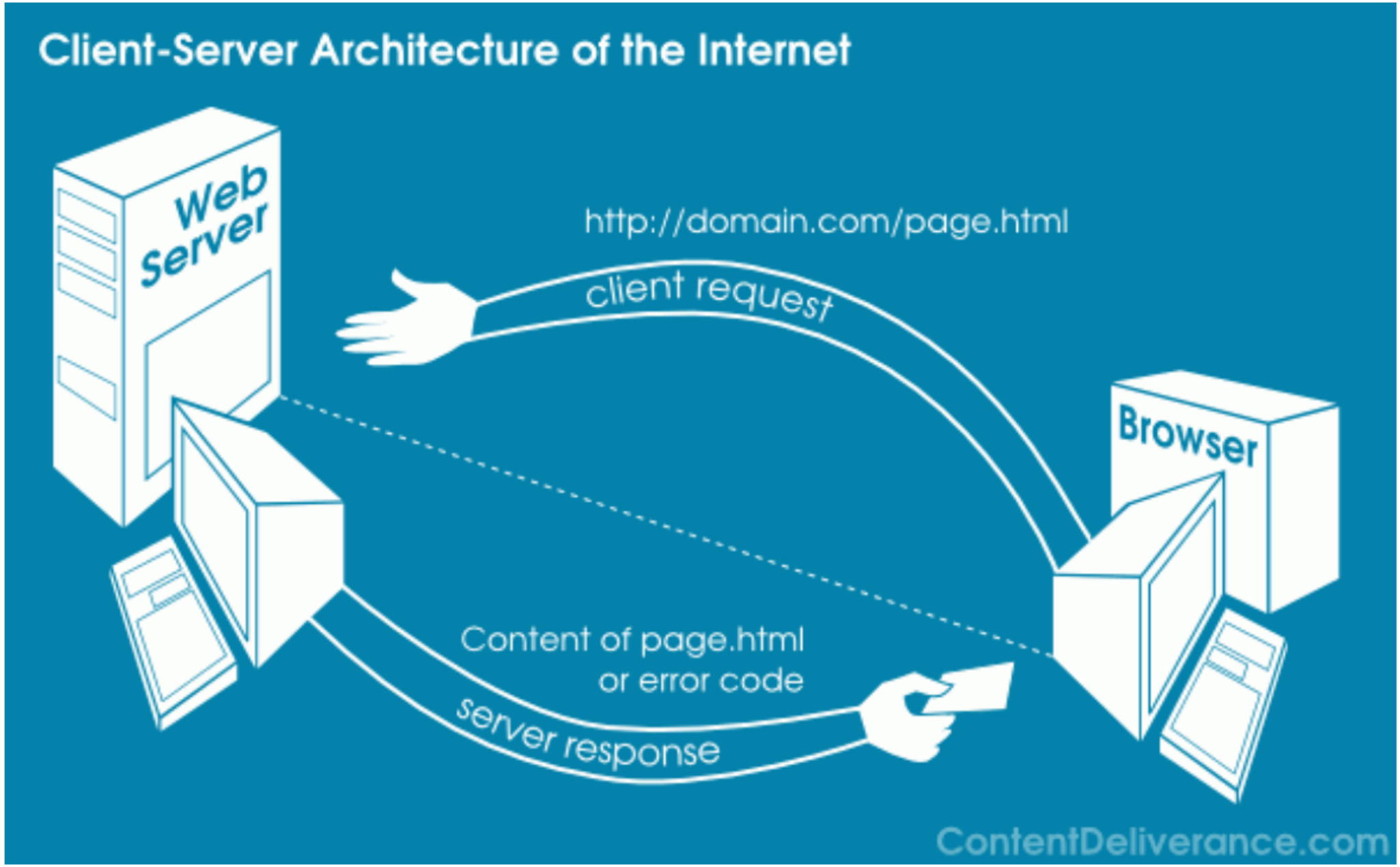
Le protocole HTTP permet le transfert de fichiers localisés grâce à une URL (notamment ceux aux format HTML et CSS), entre un navigateur (le client) et un serveur web. Cette opération se fait en deux temps avec l'envoi d'une requête puis la réception de la réponse.

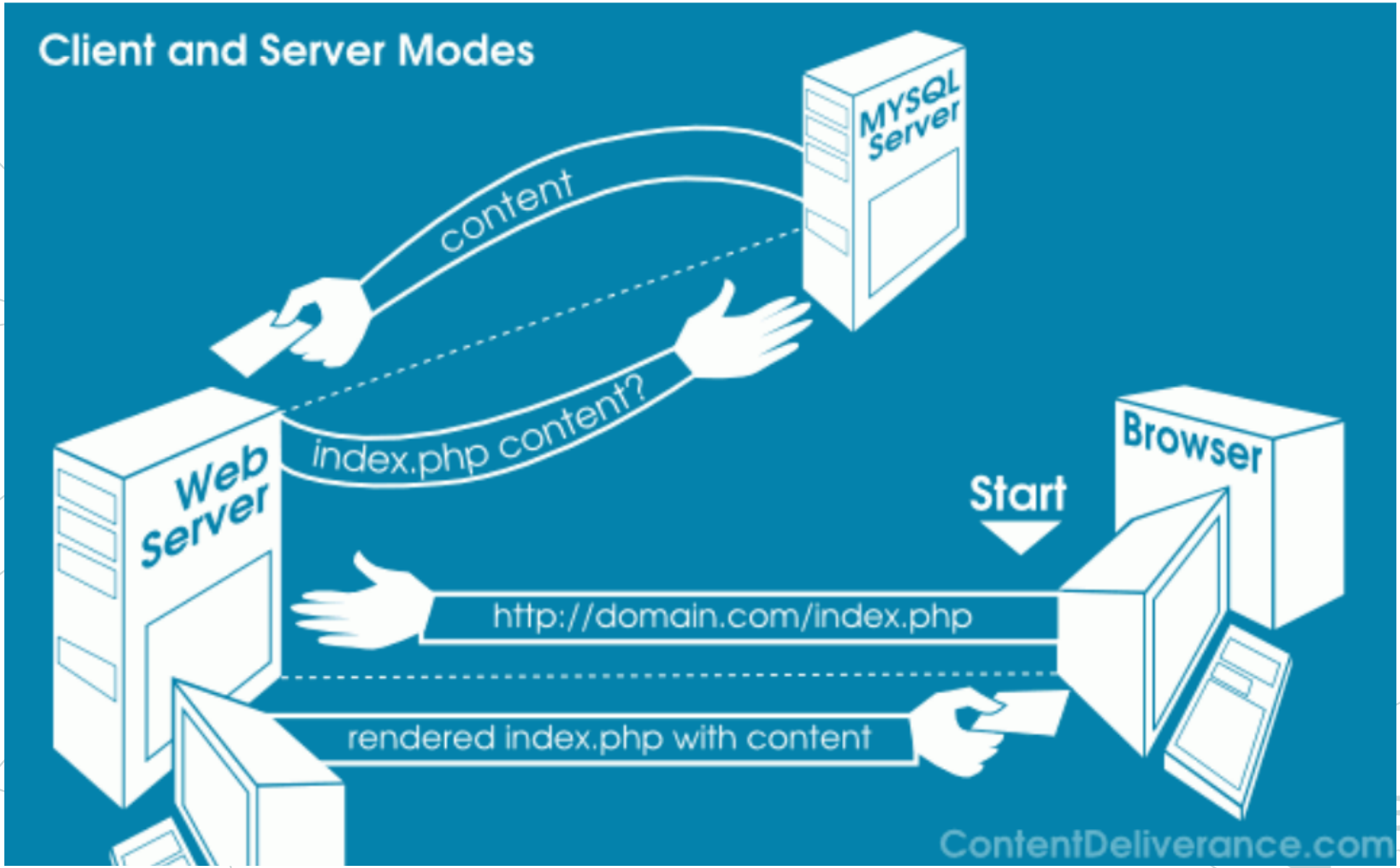


Requête HTTP



HTTP


Client-Server Architecture of the Internet
Web Server
http://domain.com/page.html
client request
Browser
Content of page.html
or error code
server response
ContentDeliverance.com


Client and Server Modes
MYSQL Server
content
index.php content?
Web Server
Browser
Start
http://domain.com/index.php
rendered index.php with content
ContentDeliverance.com

Requête HTTP

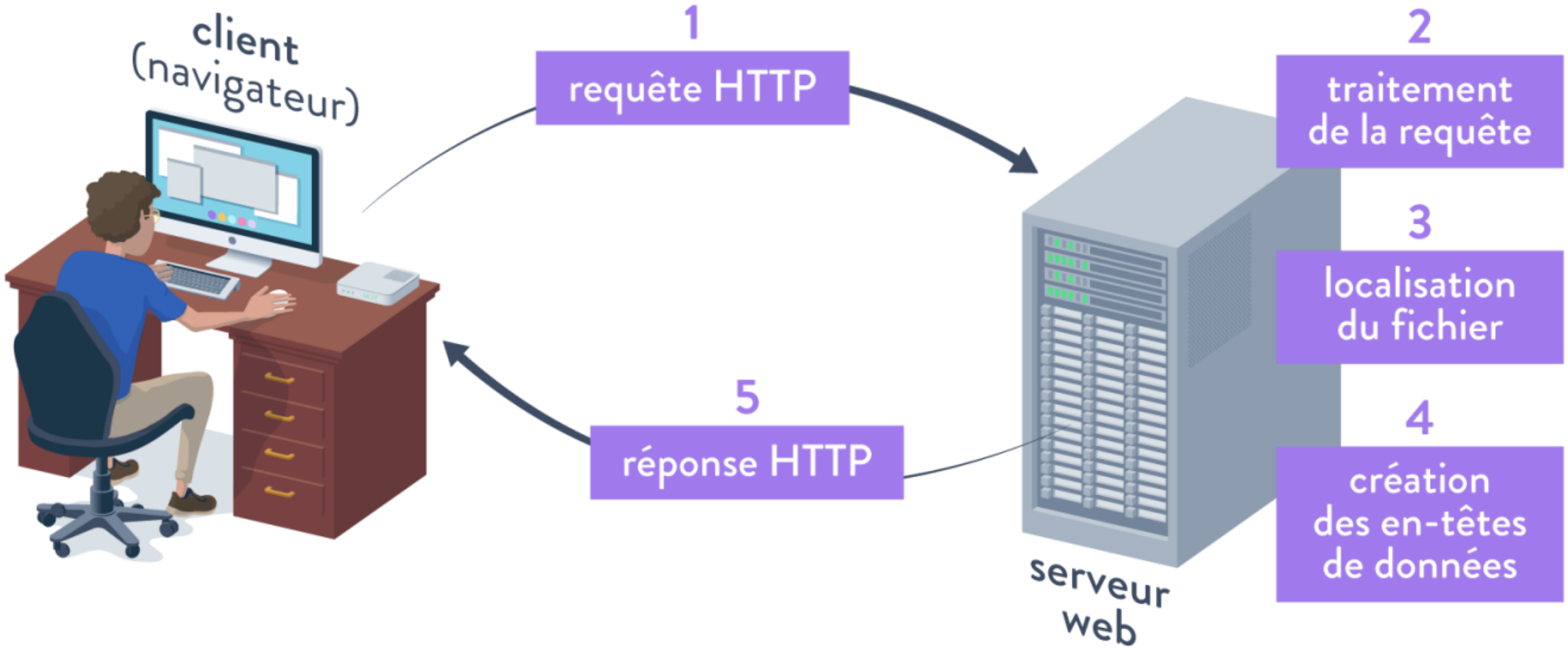
Définition

Requête HTTP :

Une requête HTTP est un ensemble de lignes envoyées au serveur par le navigateur. Elle se compose obligatoirement d'une ligne de requête, et optionnellement de champs d'en-tête et d'un corps de requête.

1. Une **ligne de requête** contient la méthode à appliquer, la ressource demandée et la version du protocole utilisé.
2. Les **champs d'en-tête** de la requête sont des lignes facultatives qui permettent de donner des informations supplémentaires sur la requête et/ou sur le client (navigateur, système d'exploitation...).
3. Le **corps** de la requête est un ensemble de lignes optionnelles séparé des lignes précédentes par une ligne vide. Il permet par exemple d'envoyer avec la méthode POST des données provenant d'un formulaire rempli par l'internaute.

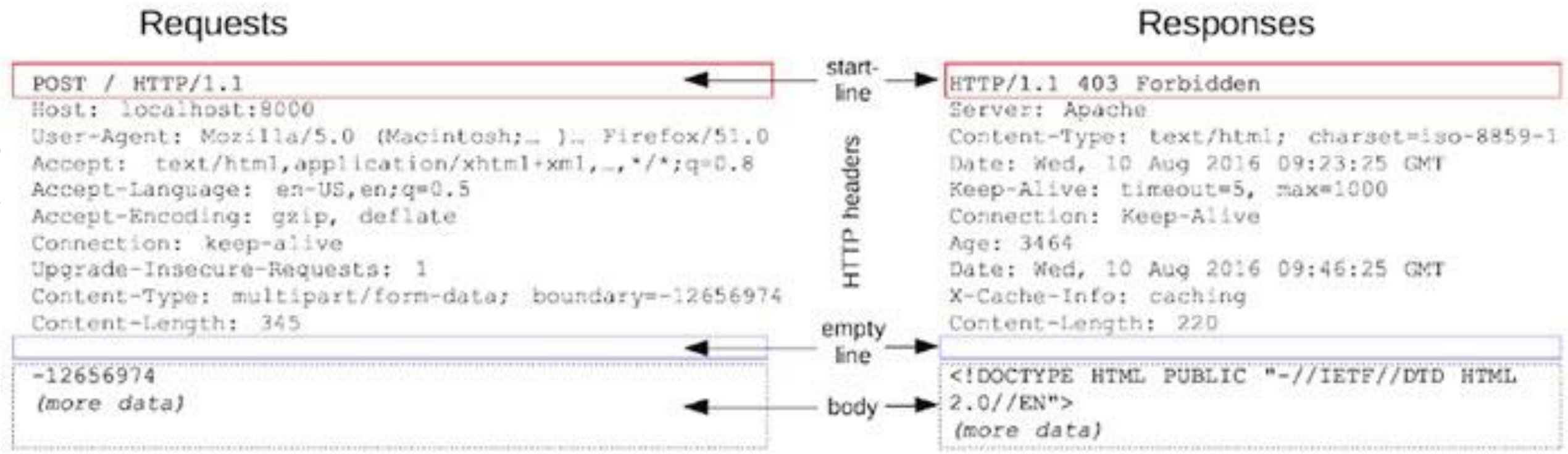
Protocole HTTP



Construction d'une requête

TBDC
Thomas BUREAU DU COLOMBIER

Contenu d'une requête HTTP



TBDC
Thomas BUREAU DU COLOMBIER

3 parties d'une requête HTTP

Ligne de requête

- Méthode
- URL
- Version

Entête de requête

- Facultatif
- Informations courtes et supplémentaire

Corps de requête

- Envoi de données plus important

TBDC
Thomas BUREAU DU COLOMBIER

3 parties d'une requête HTTP : ligne de requête

Ligne de requête

- Méthode
- URL
- Version

Entête de requête

- Facultatif
- Informations courtes et supplémentaire

Corps de requête

- Envoi de données plus important

TBDC
Thomas BUREAU DU COLOMBIER

Ligne de requête

La première ligne d'une requête HTTP permet de préciser la requête HTTP. Celle-ci est composée de :

- **La méthode de requête** utilisée qui sert à indiquer le type de requête effectuée : simple récupération de ressources, envoi de données sur le serveur, etc. ;
- **La cible de la requête** (si applicable) qui va généralement prendre la forme d'une URL ou d'un chemin absolu ;
- **La version HTTP** utilisée pour la requête (qui sert également à indiquer la version attendue pour la réponse

Ligne de requête : la méthode

GET

La méthode GET demande une représentation de la ressource spécifiée. Les requêtes GET doivent uniquement être utilisées afin de récupérer des données.

HEAD

La méthode HEAD demande une réponse identique à une requête GET pour laquelle on aura omis le corps de la réponse (on a uniquement l'en-tête).

POST

La méthode POST est utilisée pour envoyer une entité vers la ressource indiquée. Cela entraîne généralement un changement d'état ou des effets de bord sur le serveur.

PUT

La méthode PUT remplace toutes les représentations actuelles de la ressource visée par le contenu de la requête.

DELETE

La méthode DELETE supprime la ressource indiquée.

CONNECT

La méthode CONNECT établit un tunnel vers le serveur identifié par la ressource cible.

OPTIONS

La méthode OPTIONS est utilisée pour décrire les options de communications avec la ressource visée.

TRACE

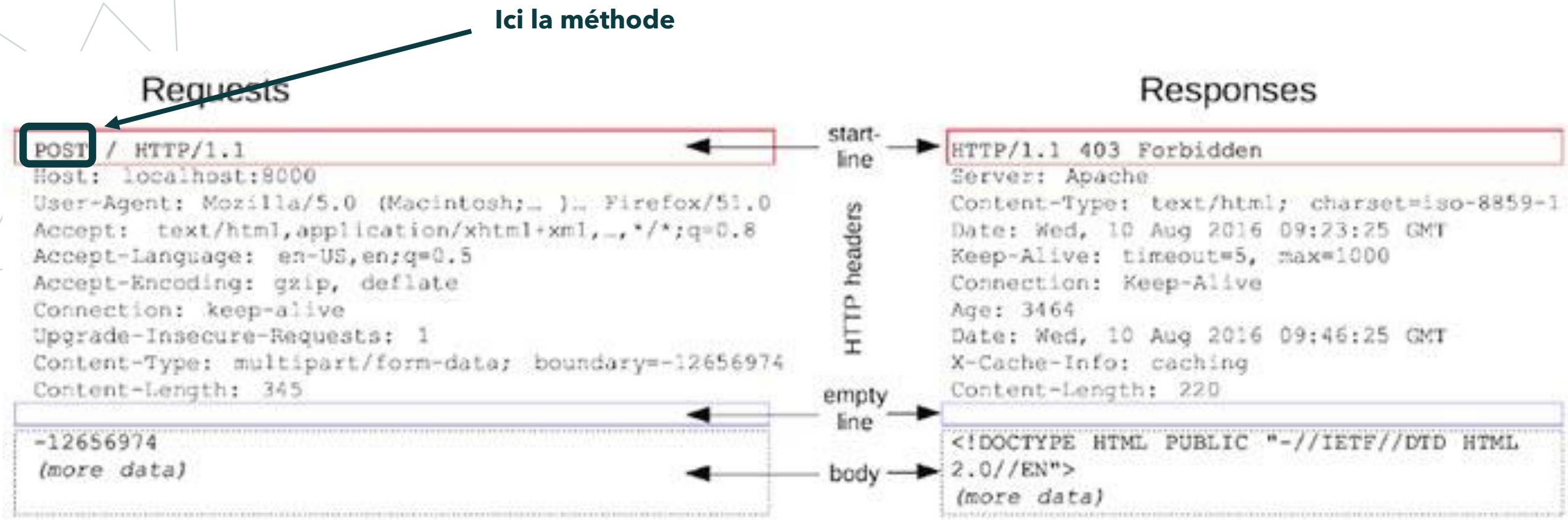
La méthode TRACE réalise un message de test aller/retour en suivant le chemin de la ressource visée.

PATCH

La méthode PATCH est utilisée pour appliquer des modifications partielles à une ressource.

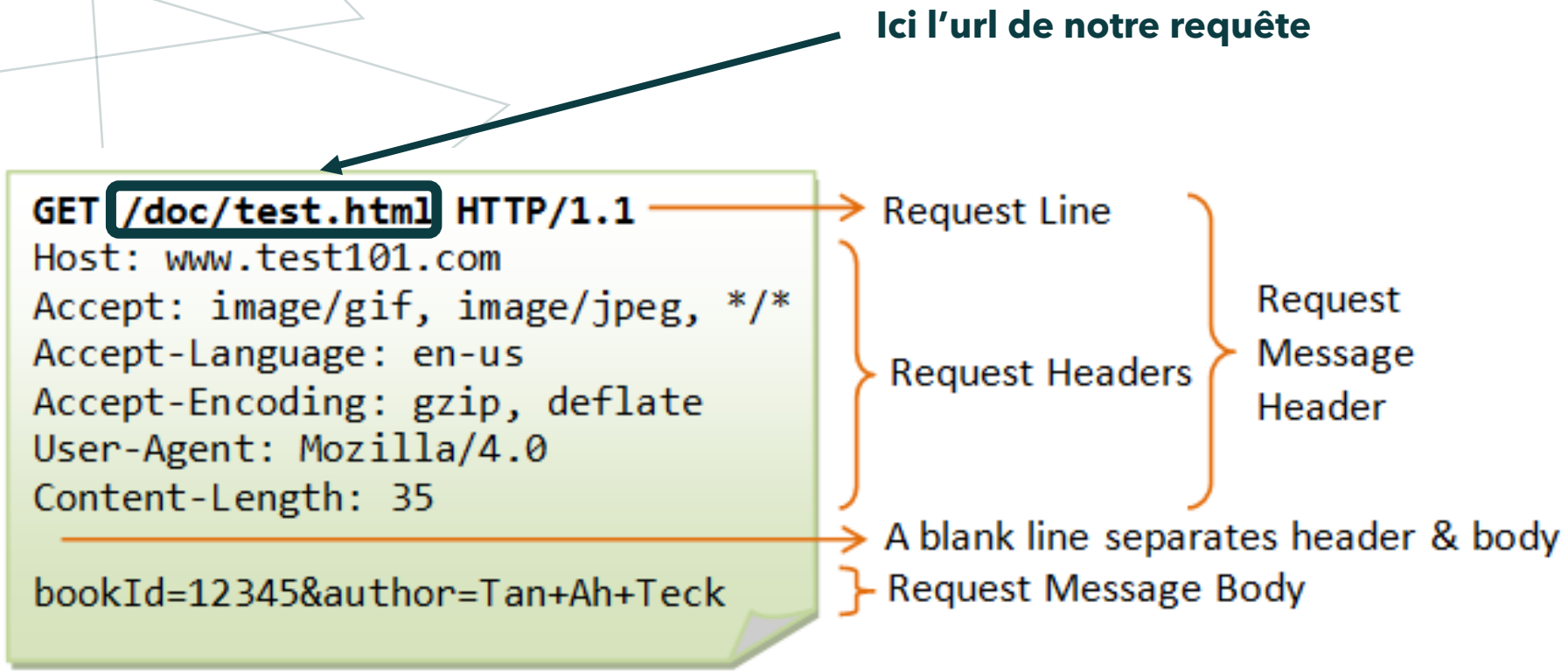
TBDC
Thomas BUREAU DU COLOMBIER

Ligne de requêtes : la méthode



Ligne de requête : l'url

- On ajoute l'url de la requête, sous forme d'url ou de chemin absolu
- Cela détermine la cible de notre requête



Ligne de requête : La version HTTP

- Deux versions majeurs : HTTP/1.1 et HTTP/2
- Depuis 2015, HTTP/2 est devenu un standard officiel

Pour plus d'informations :

https://developer.mozilla.org/fr/docs/Web/HTTP/Basics_of_HTTP/Evolution_of_HTTP

3 parties d'une requête HTTP: Entête

Ligne de requête

- Méthode
- URL
- Version

Entête de requête

- Facultatif
- Informations courtes et supplémentaire

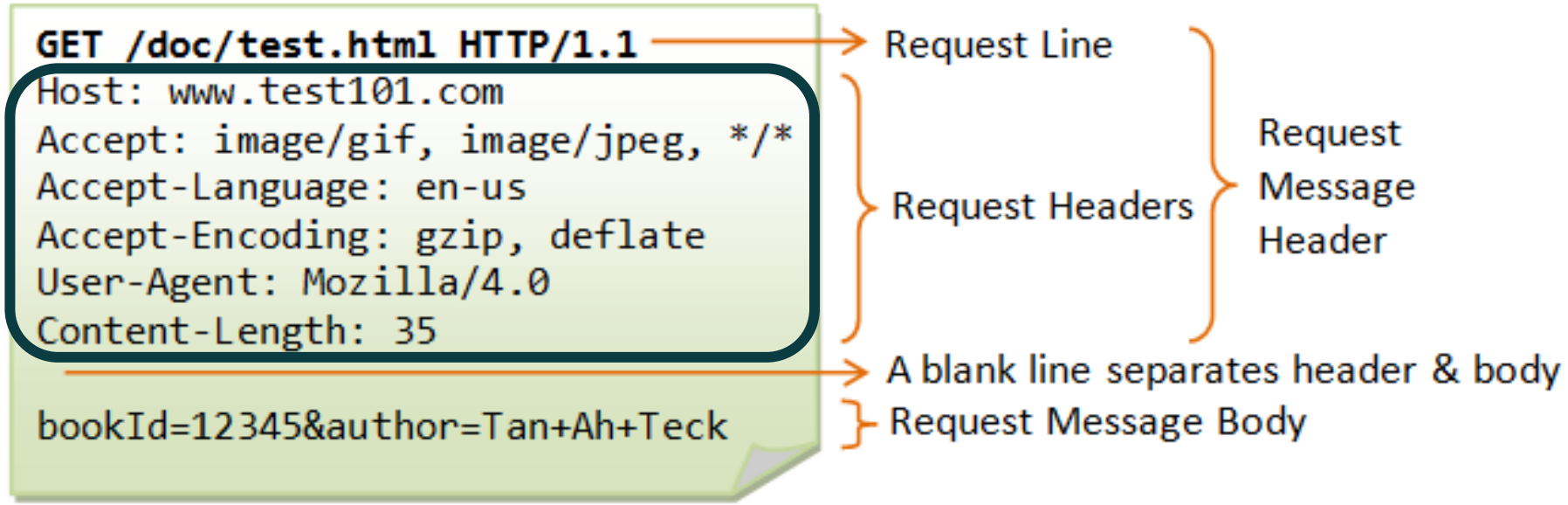
Corps de requête

- Envoi de données plus important

TBDC
Thomas BUREAU DU COLOMBIER

Entête de requête HTTP

- Permettent de préciser la requête ou d'ajouter des informations de contexte. (ex: langue préférée, comment la connexion avec le serveur doit être établie, le type et poids du corps de requête, etc.)
- Chaque entête sur une ligne



Pour plus d'informations : https://developer.mozilla.org/fr/docs/Web/HTTP/Headers

3 parties d'une requête HTTP : Corps de requête

Ligne de requête

- Méthode
- URL
- Version

Entête de requête

- Facultatif
- Informations courtes et supplémentaire

Corps de requête

- Envoi de données plus important

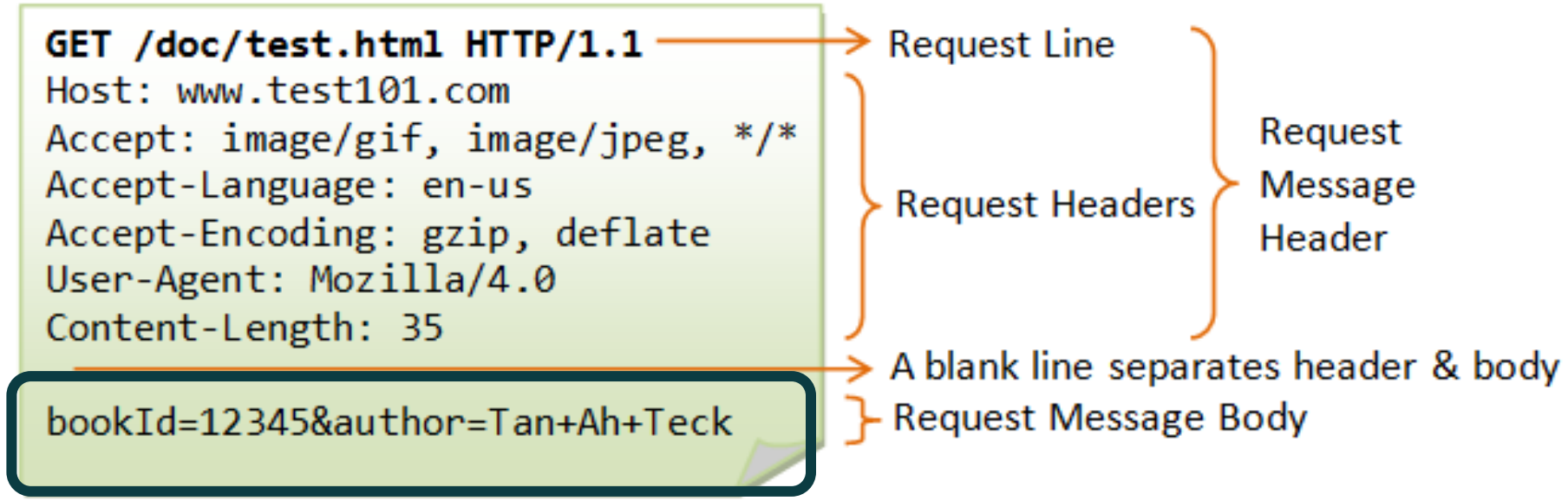
TBDC
Thomas BUREAU DU COLOMBIER

Corps de requête

- Après les lignes d'en-tête, une requête HTTP doit obligatoirement comporter une **ligne vierge** puis peut enfin comporter un **corps de requête**.
- Il permet de transmettre des données au serveur
- Le corps de requête est facultatif

Par exemple, la plupart des requêtes GET (permettant de récupérer des ressources) n'ont pas besoin d'ajouter un corps. En revanche, des requêtes comme POST qui servent à envoyer des informations au serveur vont transmettre ces informations dans le corps de la requête.

Corps de requête



Corps de requête

Corps du message (payload) : contenu proprement dit

Le corps est vide pour les requêtes GET et DELETE, non vides pour les requêtes PUT et POST.

Des en-têtes (optionnels) accompagnent fréquemment le corps et ensemble forment ce qu'on appelle l'entité du message.

- **content-type** : type du contenu exprimé sous la forme d'un type "Internet Media Type" (dit, MIME (Multipurpose Internet Mail Extensions)

  ```
  type/sous-type
  ```

  Exemple : **application/xml**, **application/json**

  Cf. la classe **javax.ws.rs.core.MediaType**.

- **content-length** : longueur du contenu

Ces informations doivent être cohérentes avec le contenu.

Différence Get et Post

GET	POST
1) In case of Get request, only **limited amount of data** can be sent because data is sent in header.	In case of post request, **large amount of data** can be sent because data is sent in body.
2) Get request is **not secured** because data is exposed in URL bar.	Post request is **secured** because data is not exposed in URL bar.
3) Get request **can be bookmarked.**	Post request **cannot be bookmarked.**
4) Get request is **idempotent** . It means second request will be ignored until response of first request is delivered	Post request is **non-idempotent.**
5) Get request is **more efficient** and used more than Post.	Post request is **less efficient** and used less than get.

```
GET/RegisterDao.jsp?name1=value1&name2=value2
```

```
POST/RegisterDao.jsp HTTP/1.1
Host: www. javatpoint.com
name1=value1&name2=value2
```

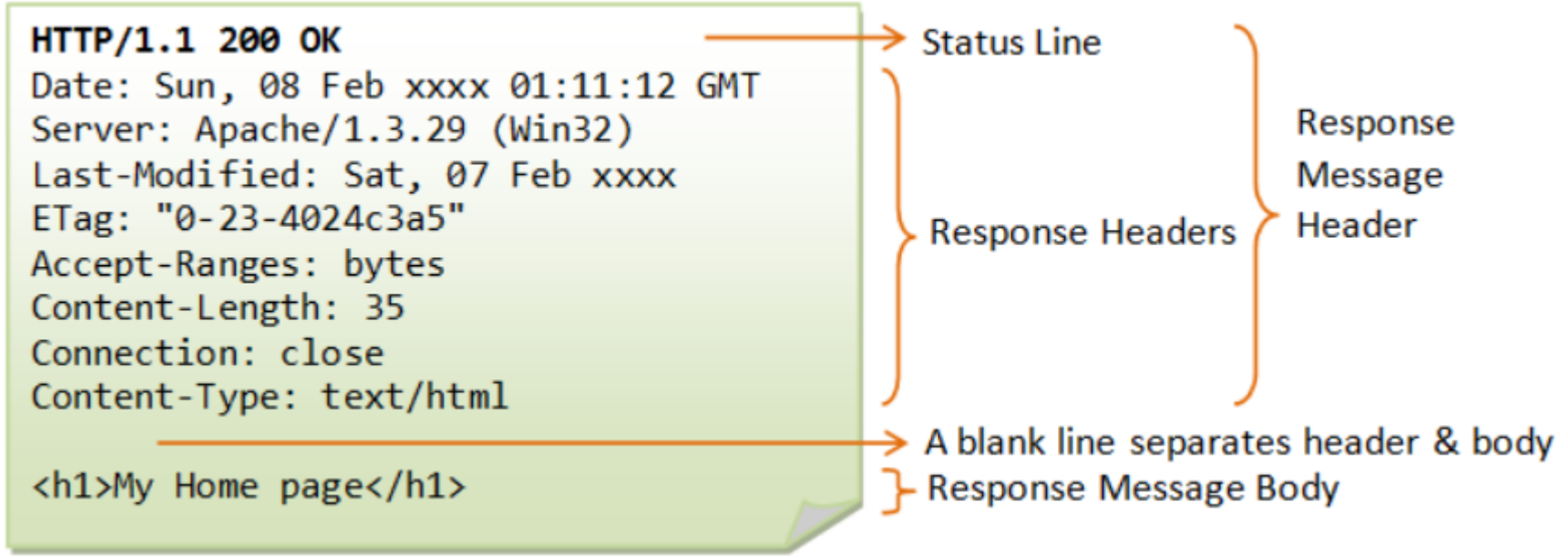
Retour de requête

TBDC
Thomas BUREAU DU COLOMBIER

Réponse HTTP

Un peu le même fonctionnement que la requête HTTP

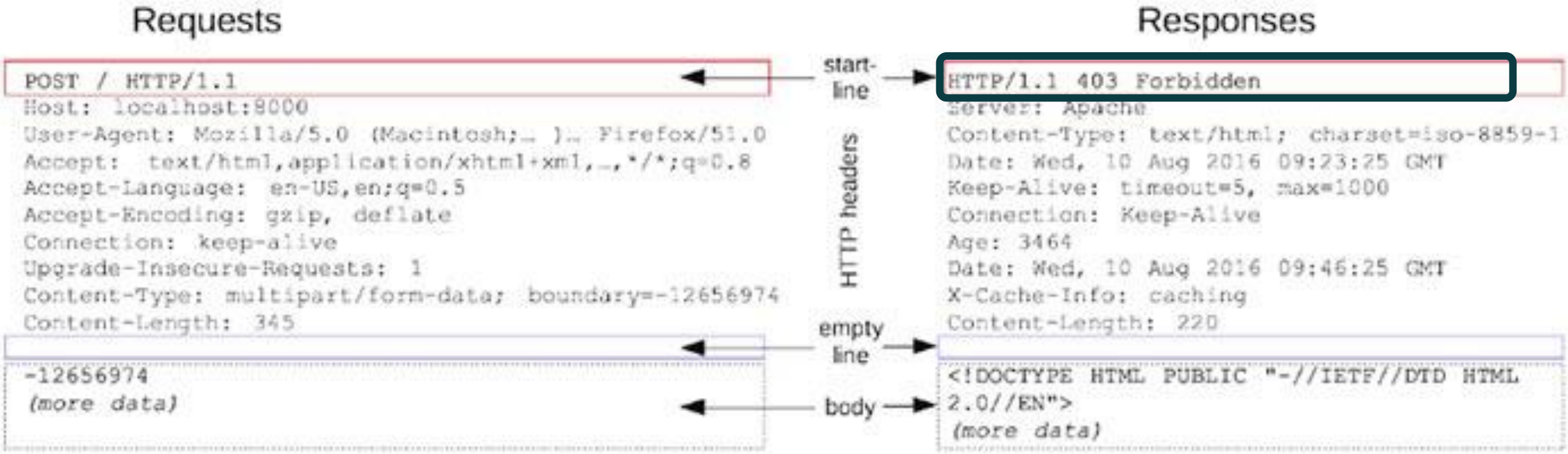
- Ligne de statut
- En-tête de réponse
- Corps de réponse



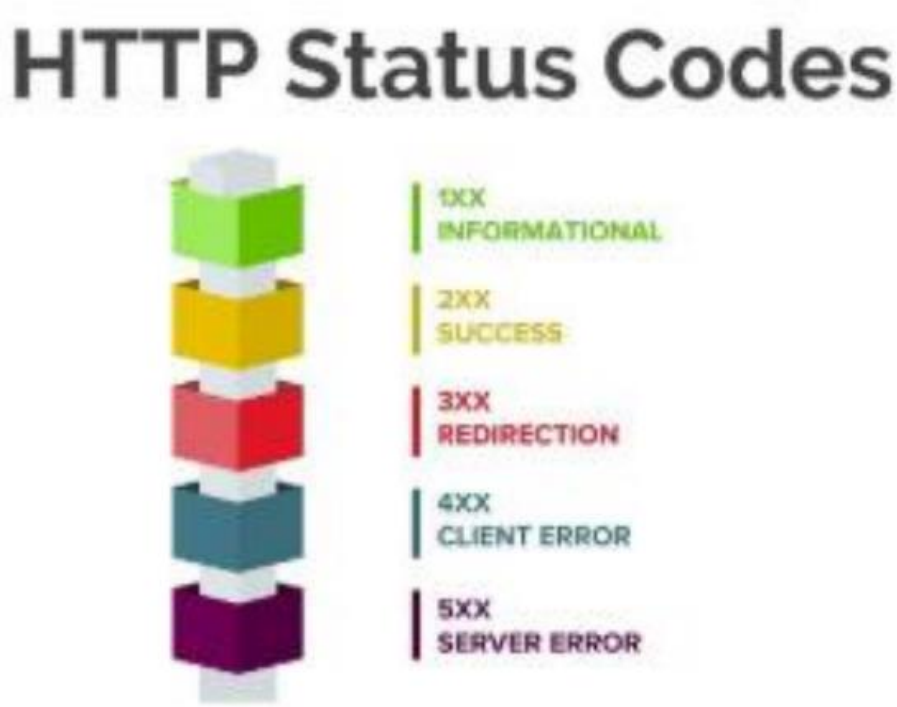
TBDC
Thomas BUREAU DU COLOMBIER

Réponse HTTP : Ligne de statut

- Version du protocole (HTTP/1.1 ou HTTP/2)
- Code de statut HTTP
- Court texte lié au code de statut



Les codes HTTP



Code	Description	Code	Description
200	OK	400	Bad Request
201	Created	401	Unauthorized
202	Accepted	403	Forbidden
301	Moved Permanently	404	Not Found
303	See Other	410	Gone
304	Not Modified	500	Internal Server Error
307	Temporary Redirect	503	Service Unavailable

Listing codes HTTP

200 OK

Le code de statut 200 (OK) indique que la requête a été traitée avec succès. La réponse du serveur est donc envoyée sans problème.

301 Moved Permanently

Le statut code HTTP 301 (Moved Permanently) indique que la ressource demandée n'est plus disponible à l'URL indiquée dans la requête mais a été déplacée vers une autre URL (qui est alors indiquée dans un en-tête Location.

Lorsque le client reçoit une réponse avec un code 301, il va généralement effectuer une nouvelle requête avec la nouvelle URL fournie pour accéder à la ressource initialement demandée.

En tant que développeur, vous serez amené à paramétrer des redirections lors de changements d'URL. En effet, utiliser des redirections 301 permet notamment de conserver les liens gagnés sur l'URL d'origine. Nous reviendrons sur les redirections plus tard dans ce cours.

302 Found et 307

Les codes de statut HTTP 302 (Found) et 307 (Temporary Redirect) servent tous les deux à indiquer une redirection temporaire, c'est-à-dire une redirection qui n'a pas vocation à perdurer dans le temps. On va utiliser les redirections temporaires lors de maintenances notamment.

304 Not Modified

Le code de statut 304 (Not Modified) permet à un serveur d'indiquer à un client que la ressource demandée n'a pas été modifiée depuis la dernière visite et que celui-ci peut donc utiliser une version placée dans son cache local plutôt que de télécharger à nouveau la ressource à partir du serveur.

Cela permet de réduire la latence et permet donc à l'utilisateur d'obtenir la ressource demandée plus rapidement.

403 Forbidden

Le code de statut 403 (Forbidden) indique que le serveur a bien compris la requête mais refuse de la compléter. Ce statut peut être renvoyé suite à un certain nombre de tentative de connexion avec des mauvais identifiants par exemple.

404 Not Found

Le code de statut 404 (Not Found) indique que le serveur n'a pas trouvé la ressource demandée et ne peut donc pas compléter la requête.

500 Internal Server Error

Le code de statut 500 (Internal Server Error) indique que le serveur a rencontré une condition inattendue qui l'a empêché de répondre à la demande. Le problème n'est cette fois-ci plus situé côté client mais bien côté serveur.

https://www.pierre-giraud.com/http-reseau-securite-cours/requete-reponse-session/

TBDC
Thomas BUREAU DU COLOMBIER

Header réponse HTTP

- HTTP/1.1 est la version valide du protocole HTTP.
- 200 OK est le code de statut. Il indique que le serveur a reçu, compris et accepté la demande.
- Content-Encoding et Content-Type fournissent des informations sur le type de fichier.
- Age, Cache-Control, Expires, Vary et X-Cache font référence à la mise en cache du fichier.
- Etag et Last-Modified sont utilisés pour le contrôle de la version du fichier livré.
- Le terme « server » désigne le logiciel du serveur Web.
- Content-Length est la taille du fichier en octets.

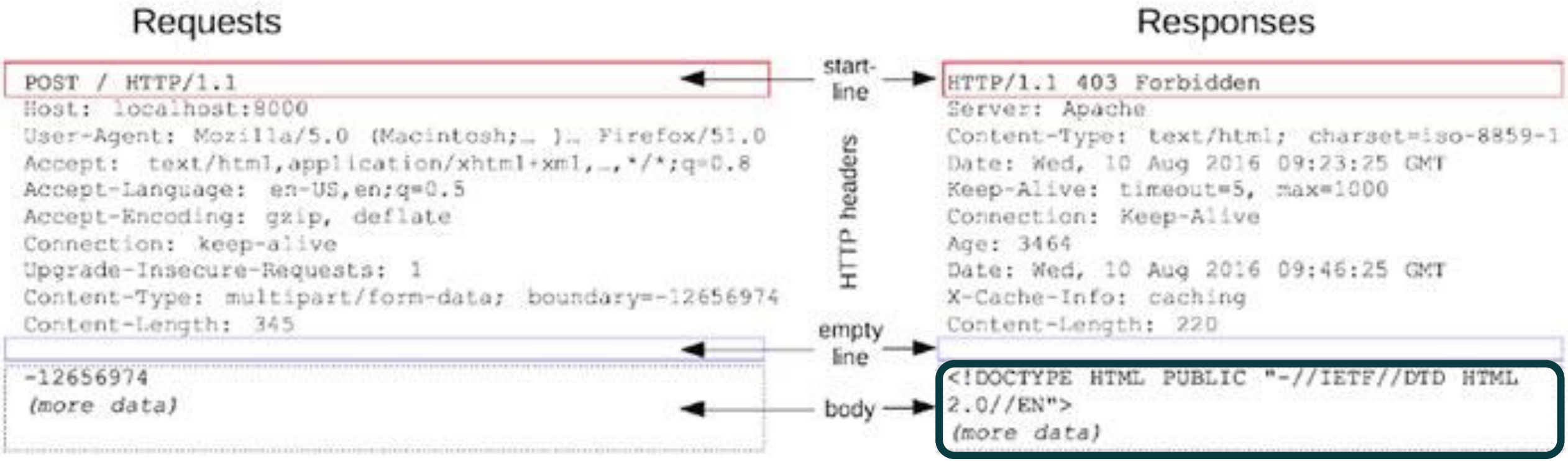
Request URI: http://www.example.com

```
HTTP/1.1 200 OK
Content-Encoding: gzip
Age: 521648
Cache-Control: max-age=604800
Content-Type: text/html; charset=UTF-8
Date: Fri, 06 Mar 2020 17:36:11 GMT
Etag: "3147526947+gzip"
Expires: Fri, 13 Mar 2020 17:36:11 GMT
Last-Modified: Thu, 17 Oct 2019 07:18:26 GMT
Server: ECS (dcb/7EC9)
Vary: Accept-Encoding
X-Cache: HIT
Content-Length: 648
```

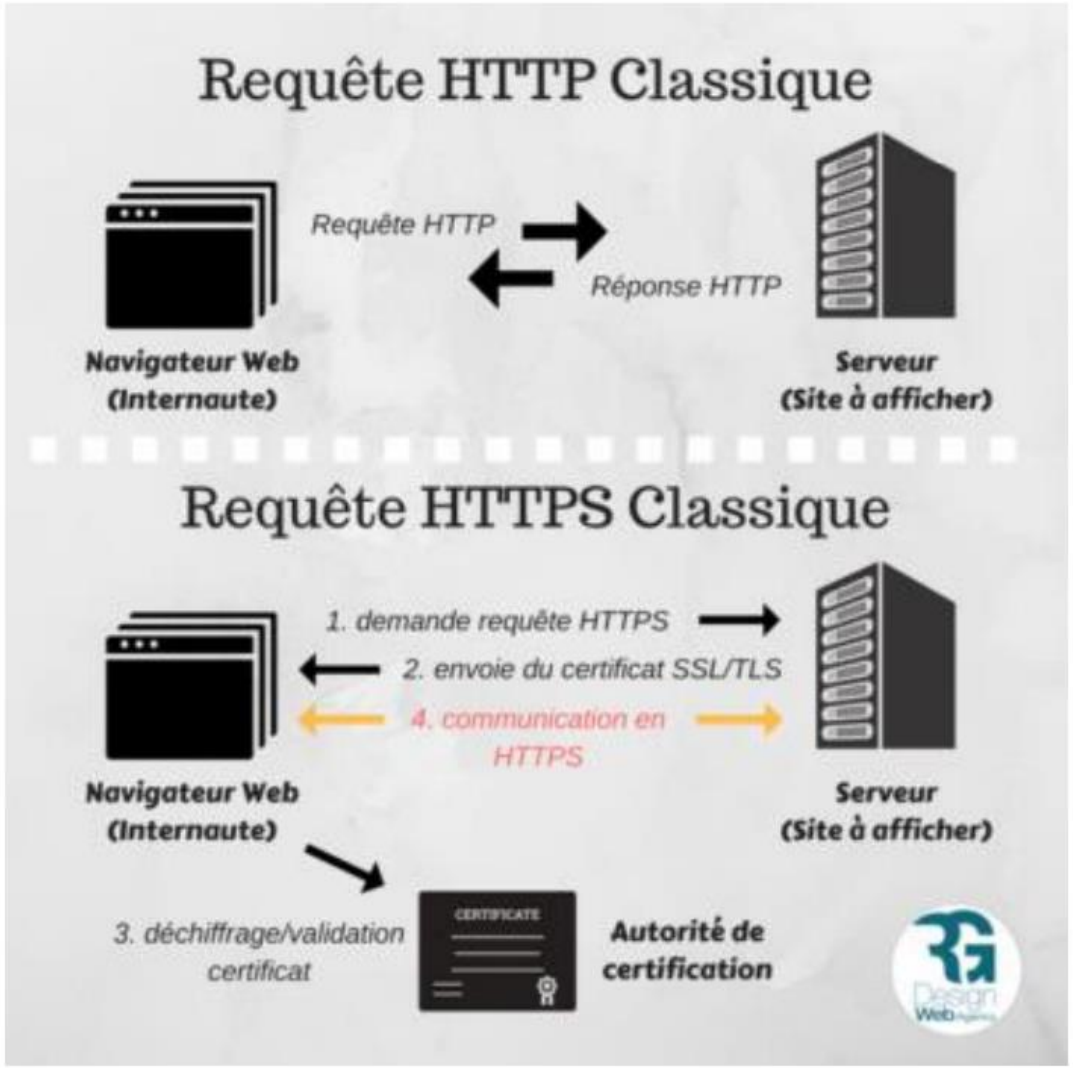

Liste des entêtes (request et response) : https://www.ionos.fr/digitalguide/hebergement/aspects-techniques/http-header/

Corps de réponse

- Facultatif
- Peut-être de différents types
- Cas courants : HTML, CSS, JS, JSON, images…

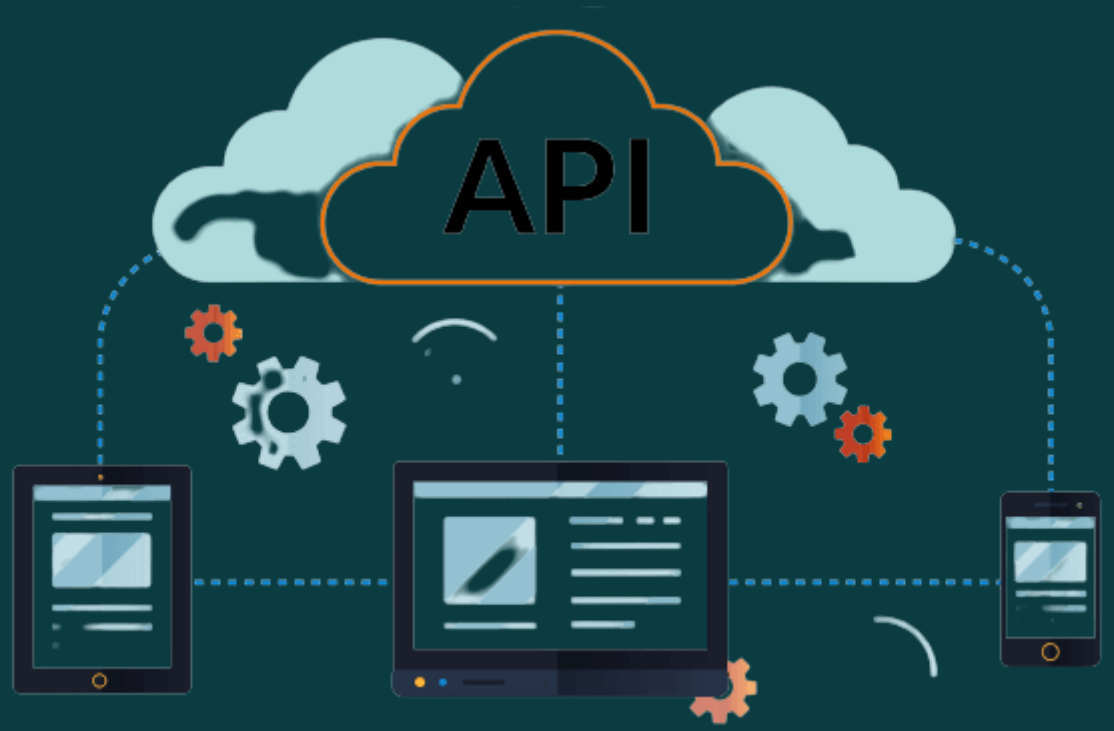


HTTPS


Requête HTTP Classique
Requête HTTP
Réponse HTTP
Navigateur Web
(Internaute)
Serveur
(Site à afficher)
Requête HTTPS Classique
1. demande requête HTTPS
2. envoie du certificat SSL/TLS
4. communication en HTTPS
Navigateur Web
(Internaute)
Serveur
(Site à afficher)
3. déchiffrage/validation certificat
CERTIFICATE
Autorité de certification


PROTOCOLE
HTTP
http://www.
https://www.
SSL
PROTOCOLE
HTTPS

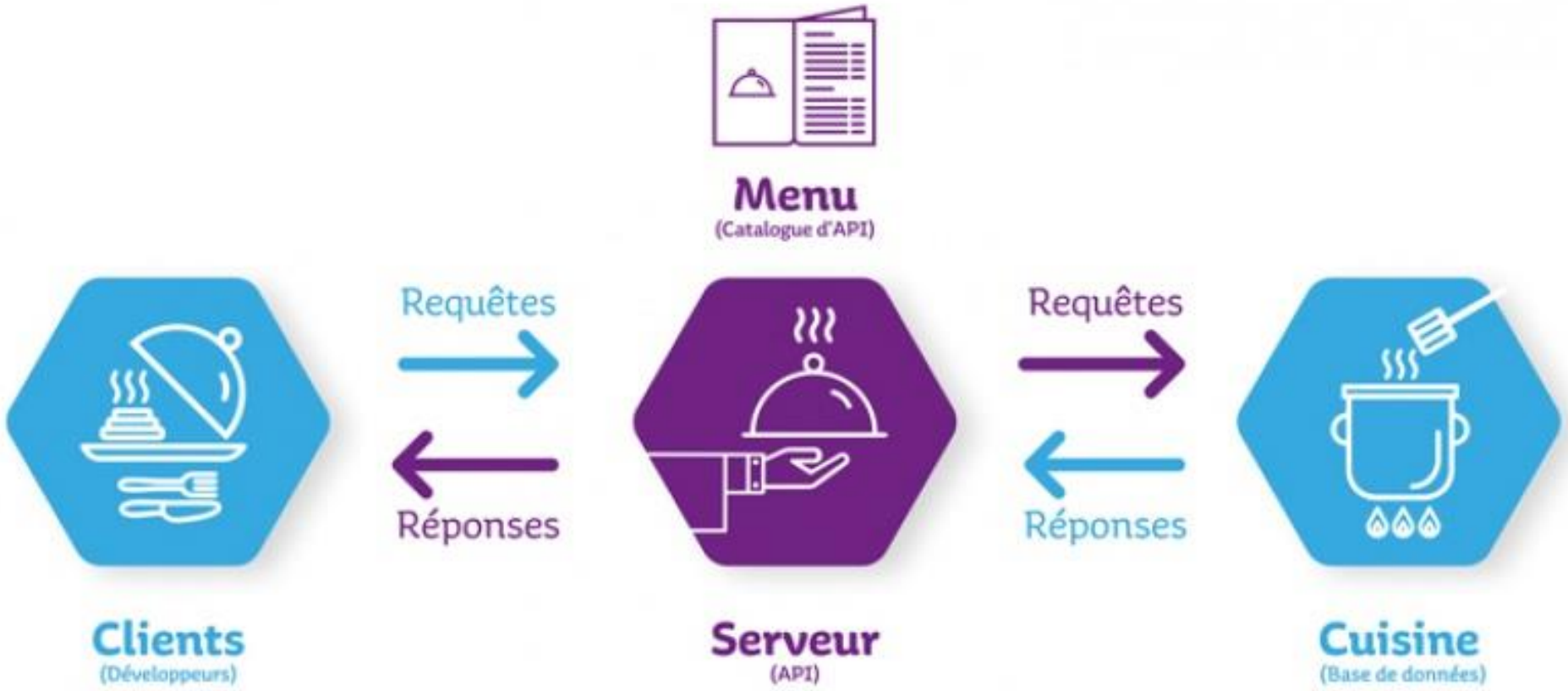
API



TBDC
Thomas BUREAU DU COLOMBIER

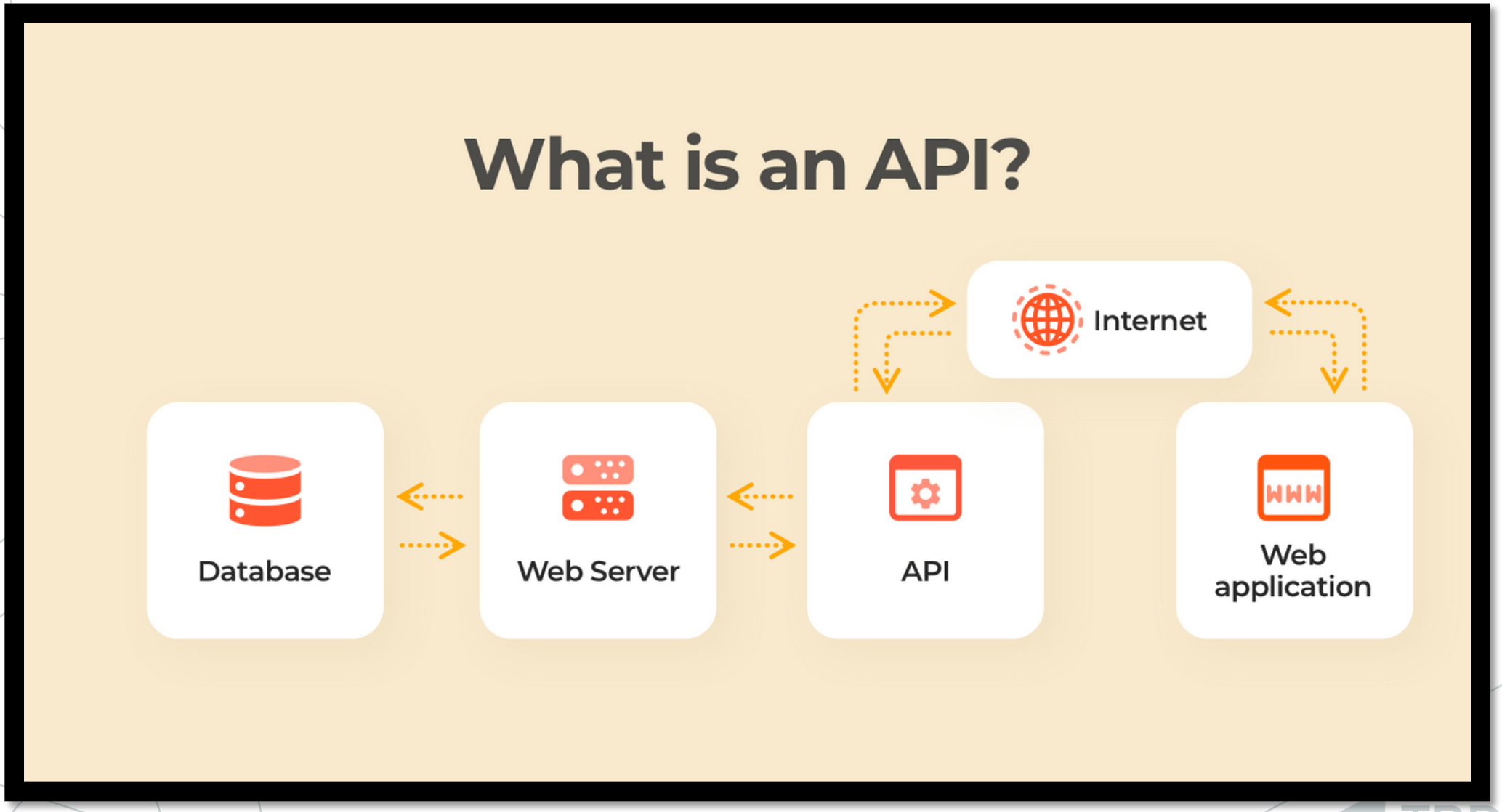
La base :Qu’est-ce qu’une API ?

Expliquer une API à vos clients

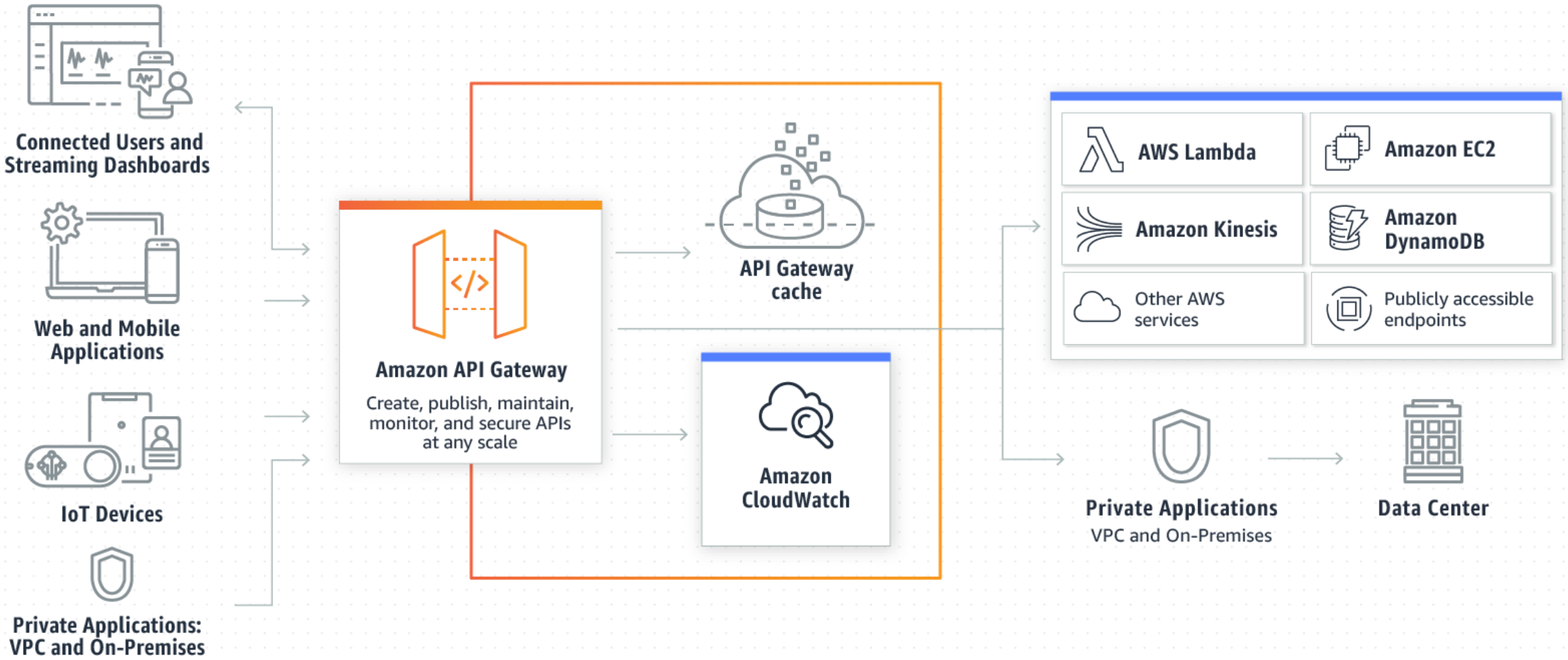


TBDC
Thomas BUREAU DU COLOMBIER

Une API


What is an API?
Internet
Database
Web Server
API
Web application
WWW

Exemple : AWS


Connected Users and Streaming Dashboards
Web and Mobile Applications
IoT Devices
Private Applications: VPC and On-Premises
Amazon API Gateway
Create, publish, maintain, monitor, and secure APIs at any scale
API Gateway cache
Amazon CloudWatch
AWS Lambda
Amazon EC2
Amazon Kinesis
Amazon DynamoDB
Other AWS services
Publicly accessible endpoints
Private Applications
VPC and On-Premises
Data Center

Définitions

Une Api est considéré souvent comme un contrat

Par exemple, l'API conçue pour un service de météo peut demander à l'utilisateur de fournir un code postal et au producteur de renvoyer une réponse en deux parties : la première concernant la température maximale et la seconde la température minimale

L’ API permet d'indiquer au système ce que vous attendez de lui, afin qu'il puisse comprendre votre demande et y répondre.

Avantages

- Médiateur entre vos services et vos utilisateurs
- Accessible pour tous les appareils connectés (un seul service pour différents clients)
- Facile à mettre en place
- Partageable avec un tiers parti extérieur à votre structure

API : REST ou SOAP ?

TBDC
Thomas BUREAU DU COLOMBIER

REST/SOAP

REST et SOAP : définition

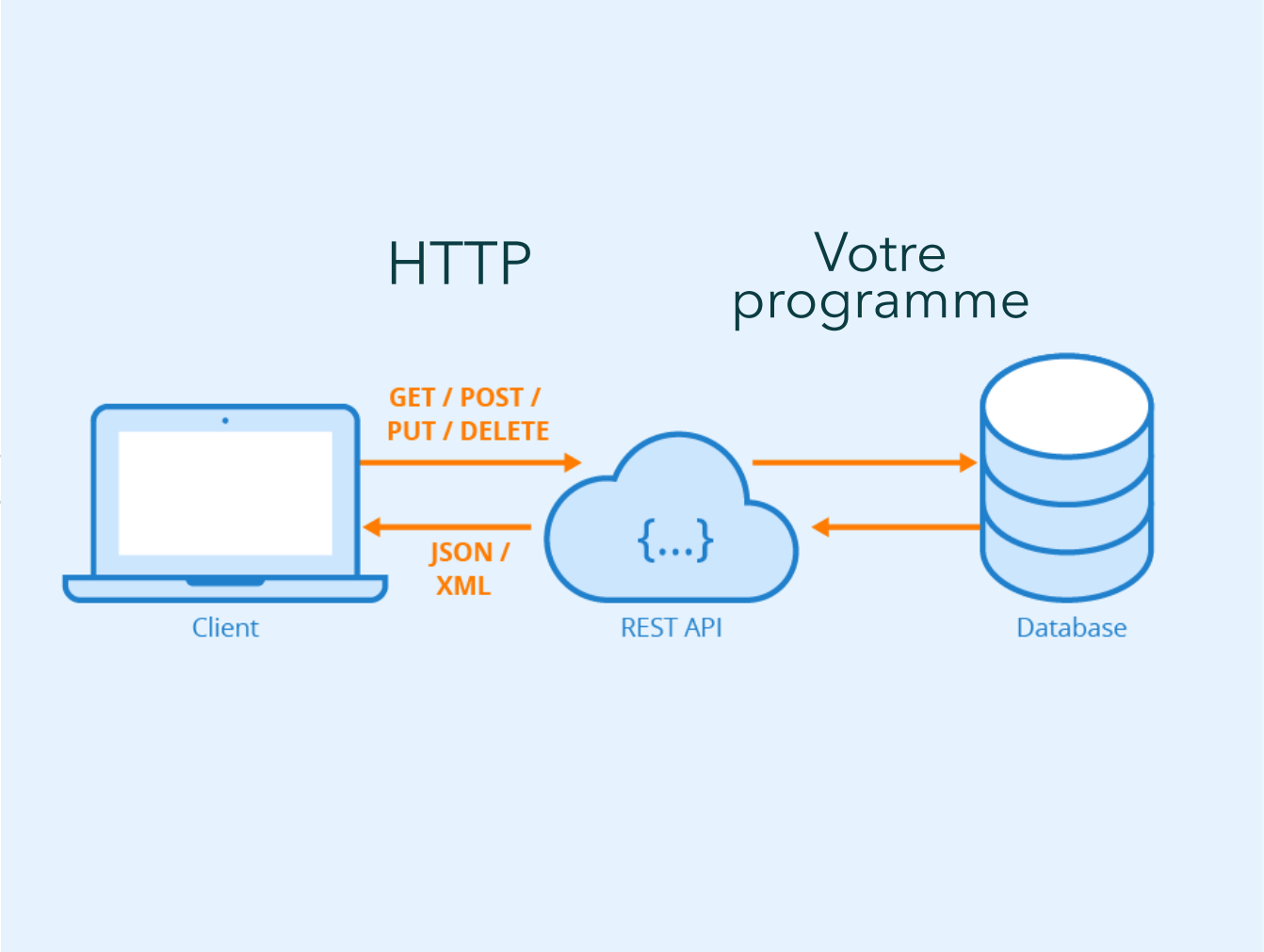
REST et SOAP sont des approches différentes de la transmission des données en ligne. Plus précisément, toutes deux définissent la manière de développer des interfaces de programmation d'application (API) qui permettent les échanges de données entre plusieurs applications web. REST (Representational State Transfer) est un ensemble de principes architecturaux. SOAP (Simple Object Access Protocol) est un protocole officiel géré par le W3C (World Wide Web Consortium). La principale différence entre les deux est que SOAP est un protocole, REST non. En général, les API suivent l'approche REST ou SOAP en fonction de leur utilisation et des préférences du développeur.

https://www.redhat.com/fr/topics/integration/whats-the-difference-between-soap-rest

REST/SOAP

#	SOAP	REST
1	A XML-based message protocol	An architectural style protocol
2	Uses WSDL for communication between consumer and provider	Uses XML or JSON to send and receive data
3	Invokes services by calling RPC method	Simply calls services via URL path
4	Does not return human readable result	Result is readable which is just plain XML or JSON
5	Transfer is over HTTP. Also uses other protocols such as SMTP, FTP, etc.	Transfer is over HTTP only
6	JavaScript can call SOAP, but it is difficult to implement	Easy to call from JavaScript
7	Performance is not great compared to REST	Performance is much better compared to SOAP - less CPU intensive, leaner code etc.

API REST

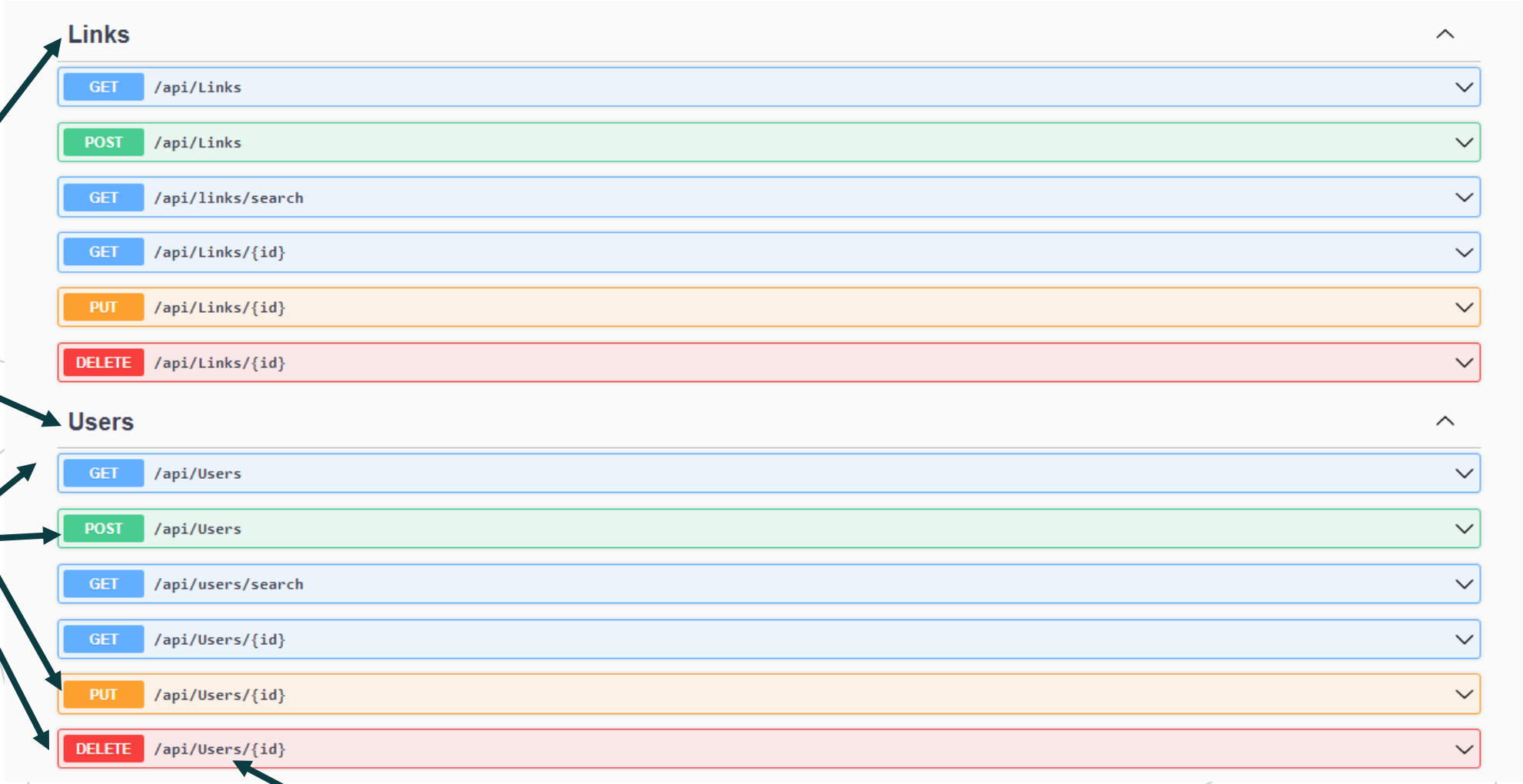


Définition d'API

Deux contrôleurs

Méthode HTTP

URL



Définition d'api

Body attendu

Codes de retours possibles

POST /api/Links

Parameters | Try it out

No parameters

Request body | application/json

Example Value | Schema

```
{
  "idLink": 0,
  "title": "string",
  "description": "string",
  "link": "string",
  "idAuthor": 0
}
```

Responses

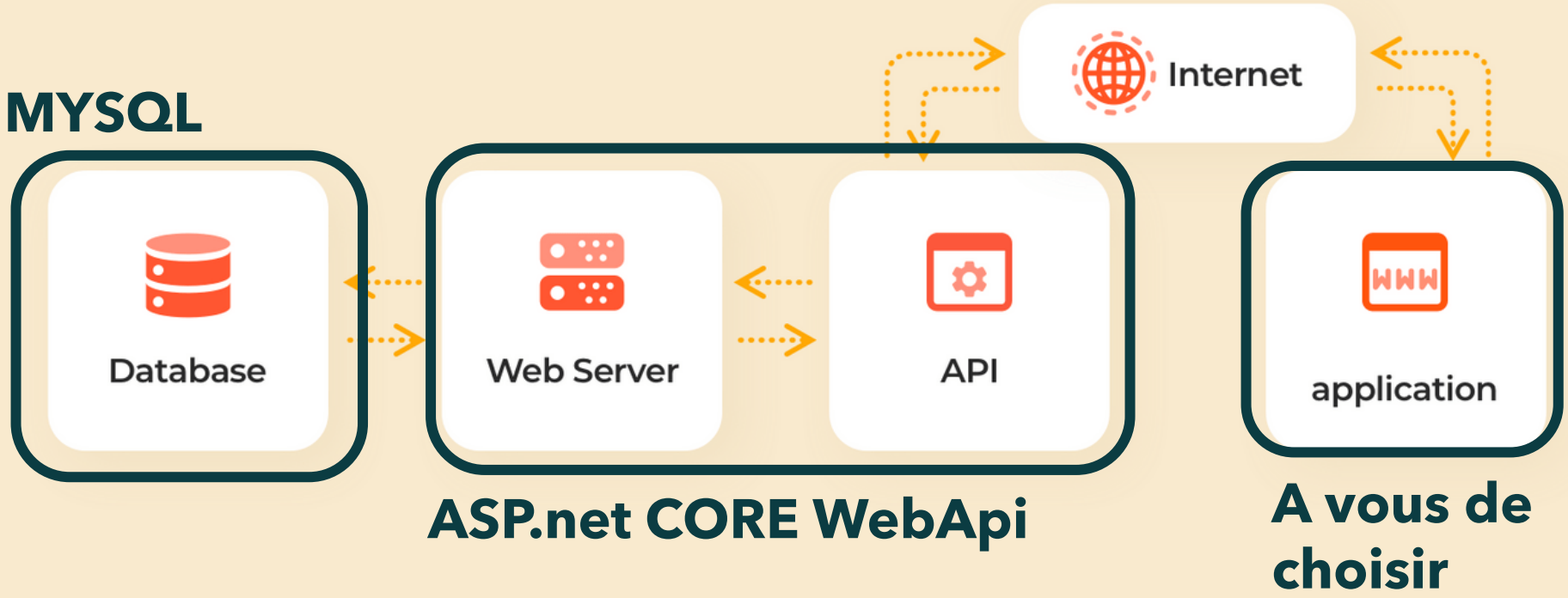
Code	Description	Links
201	Created	*No links*
400	Bad Request	*No links*

Media type: text/plain

Example Value | Schema

```
{
  "type": "string",
  "title": "string",
  "status": 0,
  "detail": "string",
  "instance": "string",
  "additionalProp1": "string",
  "additionalProp2": "string",
  "additionalProp3": "string"
}
```

Et nous on va faire quoi ?



Pause ? Après on pratique

TBDC
Thomas BUREAU DU COLOMBIER