Architectures et technologies WEB

Introduction aux technologies WEB

V01.00 17/08/2016 - MAURICE

Déroulement du cours

- *Chapitre 1 : Préambule*
- *Chapitre 1 : Les technologies WEB*
- *Chapitre 2 : La Programmation cliente*
- *Chapitre 3 : Le modèle MVC / Les templates*
- Chapitre 4 : La sécurité
- *Chapitre 5 : Les bases de PHP / Le Contrôleur*
- *Chapitre 6 : PHP 2ème partie / ORM*
- **Chapitre 7 : SOA / ROA**
- Chapitre 8 : Le référencement

Plan de la séance

- Chapitre 7 :
  - Architecture SOA
  - SOAP
  - Architecture ROA
  - Les Microservices
  - RESTFUL
  - Travaux dirigés

Architecture et développement Web

Architecture Orientée Service

Architecture Orientée Services

SOA : Service Oriented Architecture

- Une architecture SOA est une architecture logicielle s'appuyant sur un ensemble de services simples ;
- Elle fournit un cadre conceptuel de bonnes pratiques pour réorganiser le SI de l'entreprise ;
- Elle suit une approche métier formalisée en processus et découpées en services ;
- L'objectif est de décomposer une fonctionnalité en un ensemble de fonctions basiques : les services ;
- L'idée est d'obtenir une architecture logicielle globale décomposée en services et non pas en application.

Les Services

- Un service au sens SOA met à disposition un accès vers une fonction ;
- Il concrétise le lien entre la couche métier et l'implémentation du SI
- Un service est parfois une façade qui se positionne devant un composant
- Un service est une fonction qui reçoit des messages
- Un service est un composant distribué exposant des fonctions a forte valeur ajoutée
- Il propose une interface pérenne
- Il est invocable a distance
- Il est localisable

Les Web Services

- Le web service est un composant logiciel permettant la communication et l'échange de données entre applications et systèmes hétérogènes dans des environnements distribués
- Il s'agit d'un ensemble de fonctionnalités exposées sur internet ou sur intranet
- Les services offerts sont indépendants des choix technologiques

Les Web Services

- Les services Web apportent une réponse simple aux nombreux besoins d'interopérabilité.
- Peuvent être appelés à distance à travers un réseau, indépendamment du système d'exploitation et du langage de programmation utilisé.
- Par exemple, un client écrit en PHP s'exécutant sur Unix peut utiliser un service web développé en Java qui s'exécute sur Windows et inversement.
- Le protocole utilisé est nommé SOAP (Simple Object Access Protocol), il utilise XML et transmet des informations sur le réseau via HTTP.

Standards

- SOAP (Simple Object Access Protocol) est un protocole d'échange au format XML ;
- Il contient la description de l'enveloppe qui décrit le contenu des messages et la manière de les traiter ;
- SOAP est une spécification de communication entre service web reposant sur le protocole HTTP ;
- SOAP est indépendant des langages de programmation ou des systèmes d'exploitation

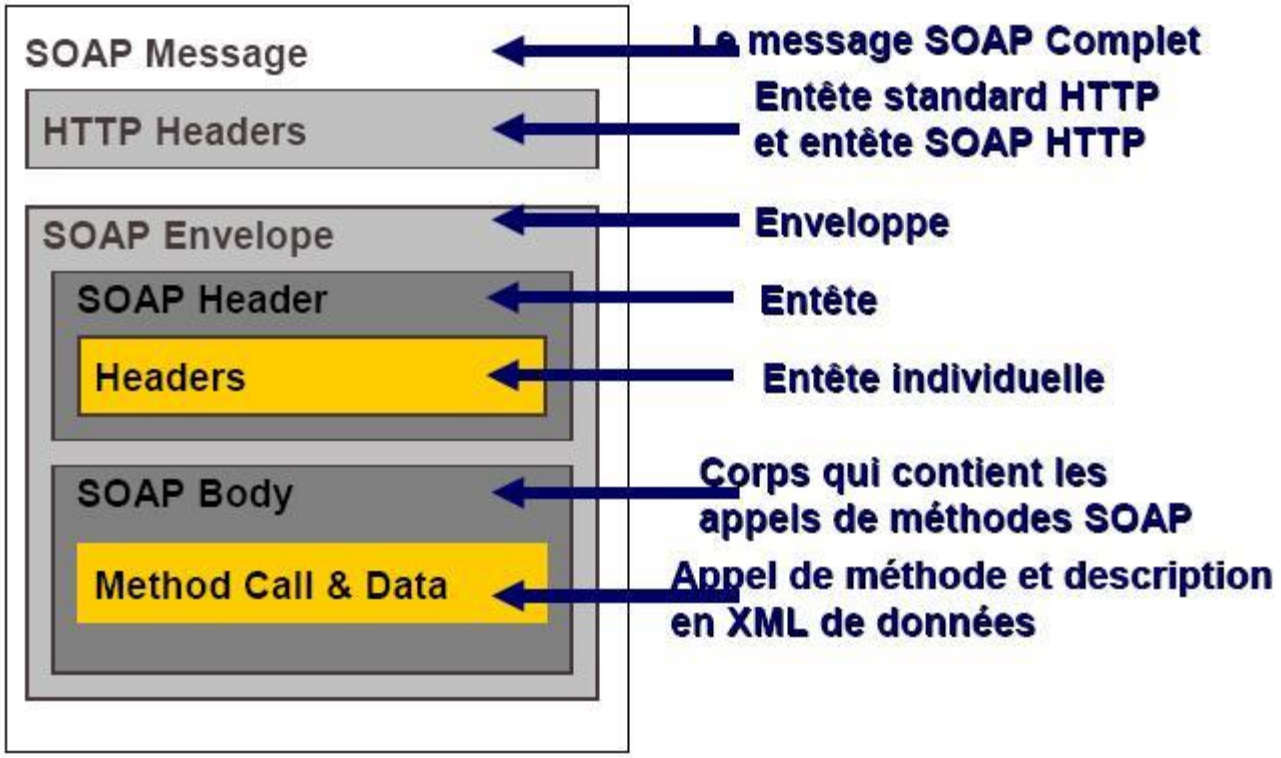
Architecture et développement Web

SOAP

SOAP

- Protocole d’échange de messages (client / serveur)
- Protocole permettant des appels de procédures à distances (RPC)
- Basé sur 2 standards
  - XML pour bien structure les messages
  - HTTP pour le transport des messages
- Concepts
  - Message = Enveloppe (Header + Body)
- Extensibilité
  - Porté sur HTTP, SMTP, …

Structure d'un Message SOAP



Infrastructures technologiques pour le commerce électronique

WSDL

Présentation

- Langage de définition de Web Services
- Basé entièrement sur XML
- Standard W3C (Initiative IBM et Microsoft)

  Actuellement WSDL 1.1
- Définition de l'interface, de l'URL et du port du Web Service.
- Utilise le système de typage de XML Schéma

Présentation

Une description WSDL :

1. Décrit le type d'un service web (méthodes, types des paramètres)

2. Décrit les aspects techniques d'implémentation d'un service web (quel est le protocole utilisé, quel est le l'adresse du service)
   Cette description permet la connexion à un service web.

Structure du fichier WSDL

Un fichier WSDL contient une description de tous les éléments nécessaires à l'appel d'un service Web SOAP :

types: décrit les types utilisés

message: décrit la structure d'un message échangé

portType: décrit un ensemble d'opérations (interface d'un service web)

operation: décrit une opération réalisée par le service web. Une opération reçoit des messages et envois des messages.

binding: décrit le lien entre un protocole (http) et un portType.

service: décrit un service comme un ensemble de ports.

port: décrit un port au travers duquel il est possible d'accéder à un ensemble d'opérations. Un port référence un Binding

Produire un WSDL

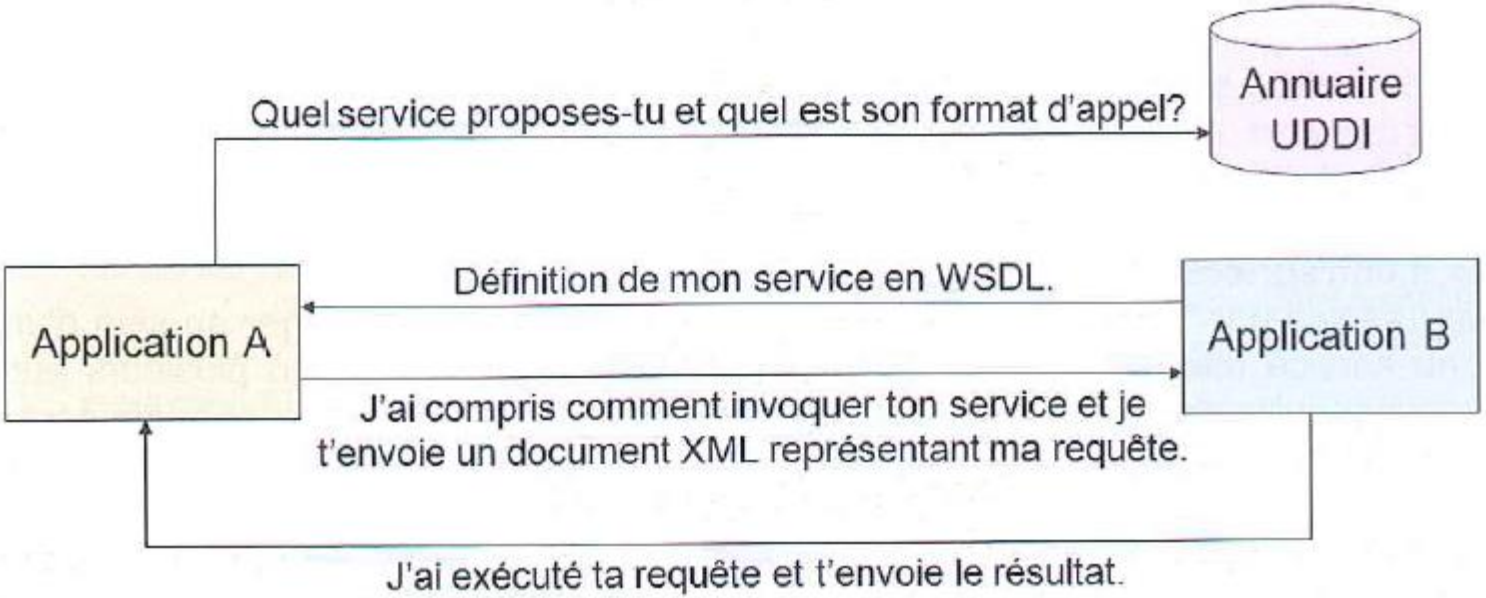
Modifiez le fichier php.ini pour y intégrer l'extension : php_soap.dll

Utilisez ZEND ou un petit outil en ligne :

http://marin.jb.free.fr/wsdl/ : Créer le fichier wsdl

```
Service : http://127.0.0.1:8887/moteur.php
Type : definition :
    string nom;
    string prenom;
    string message;
Functions prototypes :
    message getName (prenom,nom)
```

WSDL



Une fois le lien établi par Soap, l'application A reçoit un descriptif des fonctions du logiciel de gestion des achats en WSDL. Cette description comprend 6 éléments principaux: type (définition du format de données utilisé); message (information relative à une structure de document ou à des procédures); type de port (définition des opérations ou processus gérés par un ou plusieurs ports du service aussi appelés nœuds); rattachement (protocole et format de données pour un port spécifique); service (ensemble de ports reliés); port (adresse pour un rattachement).

Client

```
<?php
ini_set("soap.wsdl_cache_enabled", "0");
// lier le client au fichier WSDL
$clientSOAP = new SoapClient('Hello.wsdl');
// executer la methode getHello
echo $clientSOAP->getName('MAX','MIN');
?>
```

Serveur

```
<?php
// première étape : désactiver le cache lors de la phase de test
ini_set("soap.wsdl_cache_enabled", "0");
$serveurSOAP = new SoapServer('Hello.wsdl');
$serveurSOAP->addFunction('getName');
// lancer le serveur
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
        $serveurSOAP->handle(); }
else { echo 'désolé, je ne comprends pas les requêtes GET, veuillez
seulement utiliser POST'; }
function getName ($prenom, $nom)
{ return 'Hello ' . $prenom . ' ' . $nom; }
?>
```

Architecture et développement Web

REST

Microservices

- Caractéristiques d'un microservice :
  - L'interface doit être simple et expressive,
  - un service dispose de son propre contexte d’exécution
  - Un service gère son **propre espace de stockage**,
  - Un service doit être **inter-opérables**, quel que soit le langage de développement utilisé.,
  - Il est conseillé de privilégier une API REST/HTTP, bien qu’il n’y ait pas de consensus établi sur ce sujet.
  - Un service doit être construit, testé et déployé de manière totalement **isolée** des autres services,
  - un service peut **communiquer avec un ou plusieurs microservices**
  - Les services doivent être **faiblement couplés** entre eux

Architecture Orientée Ressources

ROA : Ressource Oriented Architecture

- Une architecture ROA est un ensemble de bonnes pratiques appliquées à REST ;
- Une ressource est référencée par une URL;
- L'URL doit être descriptive ;
- Sans état - chaque requête HTTP doit s'exécuter sans avoir connaissance des requêtes précédentes ;
- les requêtes ne doivent pas avoir d'ordre pré-défini et sont déconnectées les unes des autres ;
- Le serveur n'a jamais besoin de connaître l'état du client
  - Distinguer l'état de l'application, qui est dépendante du client, et l'état de la ressource, qui est dépendante du serveur ;
- Les interfaces sont uniformes et ces interfaces proviennent des verbes HTTP.

REST

- REST (REpresentational State Transfer) est une manière de construire une application pour les systèmes distribués.
- Le terme a été inventé par Roy Fielding en 2000.
- REST n'est pas un protocole ou un format, c'est un style d'architecture, c'est le style architectural original du Web.

REST

- REST se pose en alternative au style architectural RPC et à la plupart des cas d'utilisation de SOAP
- Les systèmes qui suivent les principes REST de Fielding sont souvent appelés RESTful.
- Même si les réponses aux requêtes sont, comme pour SOAP et XML-RPC, souvent en XML, cela n'a rien d'obligatoire.
- Des réponses JSON sont parfaitement adaptées.

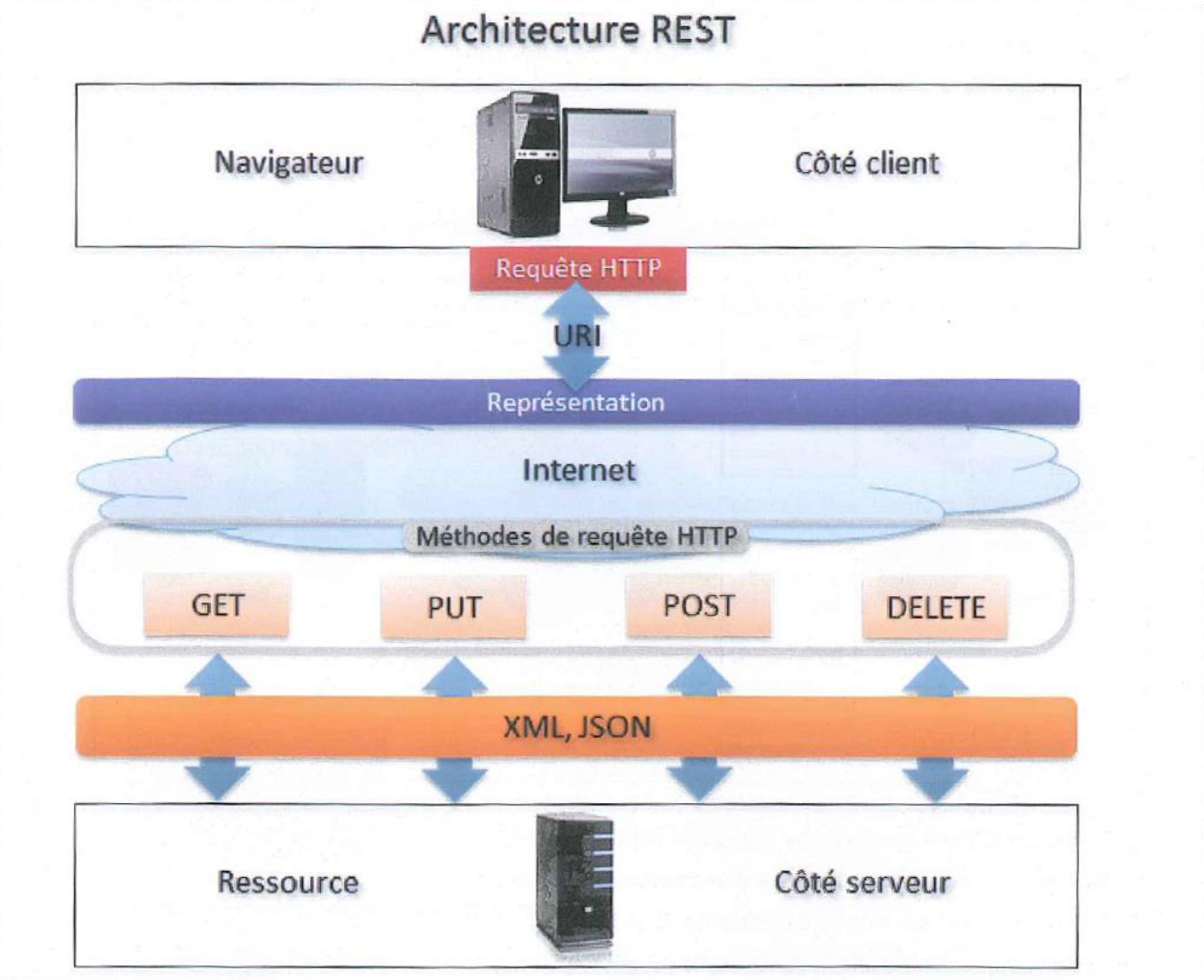
REST

- L’application est plus simple maintenir
- La consommation de mémoire est inférieure à SOAP,
- Une plus grande simplicité
- Une capacité plus grande de répondre à un grand nombre de requêtes simultanées.
- L’utilisation du protocole HTTP en tirant partie de son enveloppe
- L’utilisation d’URI comme représentant d’une ressource, permet la mise en place de serveurs cache.

REST

- L’interface entre les composants est simple et uniforme.
- En HTTP, cette interface est implantée par les *verbs* GET, PUT, POST, DELETE, ... qui permettent aux composants de manipuler les ressources de manière simple et "intuitive".
- **GET** : utilisé pour les SELECT
- **POST** : utilisé pour les INSERT
- **PUT** : utilisé pour les UPDATE
- **DELETE** : utilisé pour les DELETE

Architecture REST



SLIM

http://www.slimframework.com/

SLIM est un Framework simple pour PHP5 permettant d'implémenter des services REST.

SLIM

- Installation

```
composer require slim/slim "^3.0"
```

SLIM

Exemple simple d'un moteur RESTFUL

```
<?php
require 'vendor/autoload.php';
$app = new \Slim\App;

$app->get('/hello/{name}',
      function ($resquest,$response,$args) {
      return $response->write("Hello " . $args['name'] });

$app->run();

?>
```

SLIM

Accès client via JQUERY

```
var rootURL = "http://localhost:8887/moteur.php";
function get(name) {
    $.ajax({
        type: 'GET',
        url: rootURL + '/hello/' + name,

        success: function(data){
                    $('#test').html(data);
        }
    });
}
```

SLIM

Service Web RESTFUL en PHP

```
<?php
$app = new \Slim\App;

$app->get('/livre/{id},  'getLivre');
$app->post('/livre', 'getLivre');
$app->put('/livre/{id}', 'getLivre');
$app->delete('/livre/{id}',   'getLivre');

$app->run();
```

SLIM

Méthodes GET et ¨POST

```
function getLivre($request,$response,$args) {
    $id = $args['id'];
    // RECHERCHE
         ...
    return $response->write (json_encode($livre));
}

function addLivre($request,$response,$args) {
         $body = $request->getParsedBody();  // Parse le body
         $nom = $body['nom']; // Data du formulaire

         // AJOUT
         ...
         return $response->write ("");
}
```

SLIM

Méthodes PUT et DELETE

```
function updateLivre($request,$response,$args) {
          $id = $args['id'];
          $body = $request->getParsedBody();
          $nom = $body['nom'];
          // Mise a jour
          ...
          return $response->write ("");

}

function deleteLivre($request,$response,$args) {
    $id = $args['id'];
    // Suppression
          ...
return $response->write ("");

}
```

SLIM

Consommation d'un service RESTFUL en PHP

-> Possibilité d'utiliser également CURL

```
$url = "http://projet-web-emma-cnam.c9users.io/rest.php/client"; $data = "nom=Manu";

// Entete HTTP
$opts = array(
        'http'=>array(
                'method'=>"POST",
                'header'=>"User-Agent:Client RestFul\r\n",
                "Content-type: application/x-www-form-urlencoded\r\n",
        "Content-length: " . strlen($data)."\r\n",
        'content'=>$data));
// Flux Web
        $context = stream_context_create($opts);                                $buffer
= file_get_contents ($url,false,$context);
// decode JSON
        $retour = json_decode($buffer);
        var_dump ($retour);
```

Paypal

Paypal

- Création des comptes développeurs
  - –Créations des comptes de tests pour l'acheteur et le vendeur

https://developer.paypal.com/

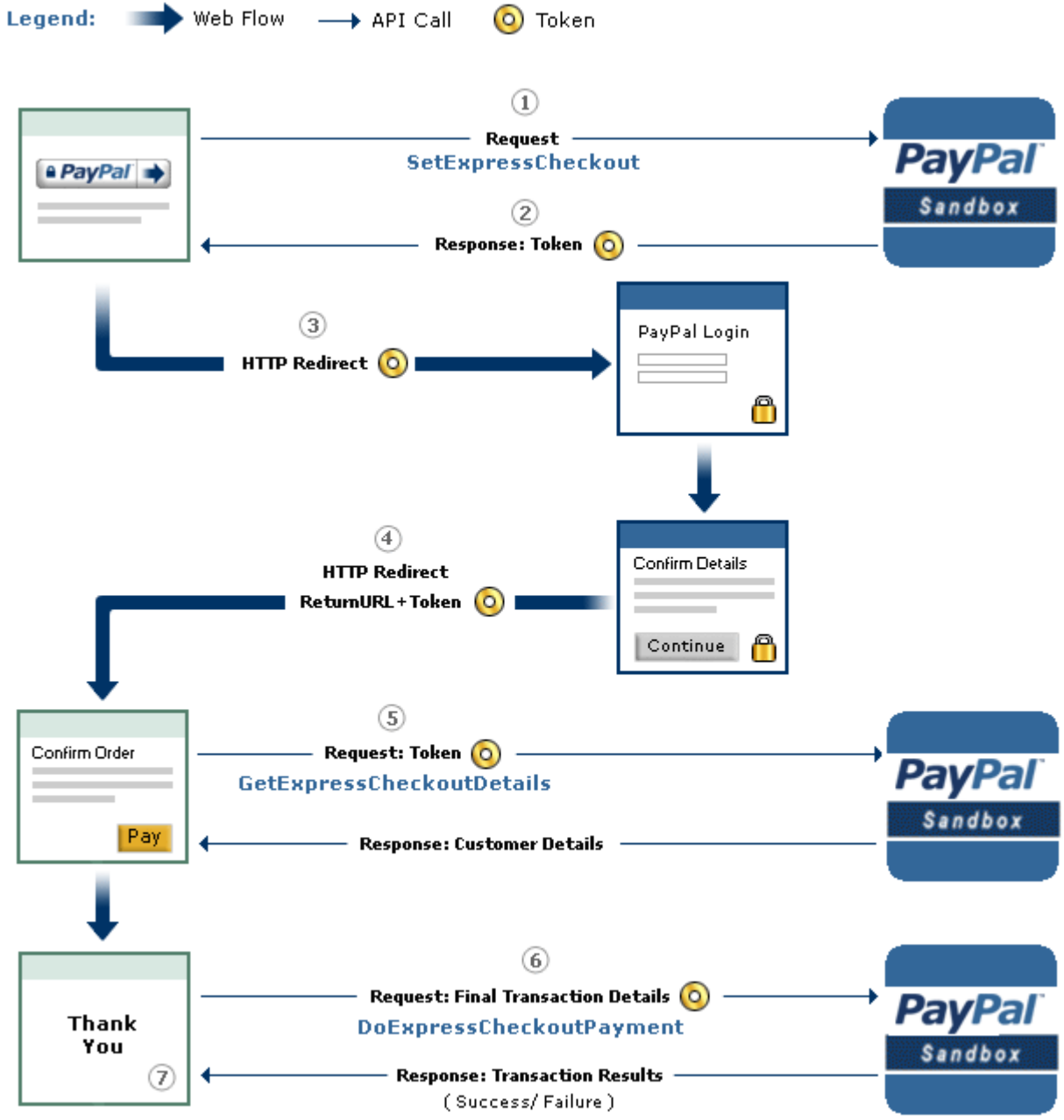
Relevez l'identifiant API, le mot de passe API ainsi que la signature de l'API.

Paypal

Effectuer un paiement express via PAYPAL :
https://developer.paypal.com/docs/classic/express-checkout/ht_ec-singleItemPayment-curl-etc/?mark=SetExpressCheckoutPayment

- Un visiteur est sur votre site et complète son panier
- Le visiteur valide son panier clique sur le bouton de paiement PayPal :
  - **1.** Le serveur de votre site web va appeler une API PayPal.
  - **2.** Le serveur Paypal retourne une chaîne de caractères alphanumérique (jeton) pour identifier le paiement.
  - **3.** Le site redirige le client sur le site de PayPal, en y associant le jeton obtenu auparavant,
  - **4.** Le client effectue son paiement sur le site de Paypal.
  - **5.** *Le serveur va envoyer une seconde demande au serveur **PayPal** via les **API** (en envoyant le jeton obtenu dans la première phase), afin de connaître les détails du client **(facultatif)**.*
  - **5b.** *Le serveur PayPal vous retourne les informations client (nom, prénom, etc.) **(facultatif)**.*
  - **6.** Le serveur renvoie une troisième et dernière demande au serveur **PayPal** via les API, afin de finaliser la transaction, et de connaître le résultat de la transaction finale.
  - **6b.** Le serveur PayPal vous informe si le paiement a été accepté ou non.

Paypa


Legend:
Web Flow
API Call
Token
1
Request
SetExpressCheckout
PayPal
Sandbox
2
Response: Token
PayPal
3
HTTP Redirect
PayPal Login
4
HTTP Redirect
ReturnURL + Token
Confirm Details
Continue
Confirm Order
Pay
5
Request: Token
GetExpressCheckoutDetails
Response: Customer Details
PayPal
Sandbox
Thank You
7
6
Request: Final Transaction Details
DoExpressCheckoutPayment
Response: Transaction Results
(Success/ Failure)
PayPal
Sandbox

Paypal

SetExpressCheckout : Création du jeton (token)

Pour obtenir le jeton nécessaire et rediriger le client sur le site de PayPal, il faut envoyer à PayPal les données suivantes :

- **USER :** correspond au nom API ;
- **PWD :** correspond au mot de passe API ;
- **SIGNATURE :** correspond à la signature de votre compte API
- **VERSION :** correspond à la version de l'API que vous souhaitez utiliser

```
https://developer.paypal.com/docs/classic/api/merchant/SetExpressCheckout_API_Operation_NVP/
```

Paypal

- **METHOD** définit la méthode paiement. **SetExpressCheckout** permet de définir un paiement et de récupérer le jeton pour rediriger votre visiteur sur le site de PayPal.
- **CANCELURL** définit l'URL sur laquelle PayPal doit rediriger le visiteur s'il annule la transaction ;
- **RETURNURL** définit l'URL en cas de traitement effectué avec succès,
- **AMT** définit le montant

METHOD=SetExpressCheckout
CANCELURL=http://www.monsite.com/cancel.php
RETURNURL=http://www.monsite.com/return.php
AMT=10.0
CURRENCYCOD=EUR

Paypal

Exemple :

```
https://api-
3t.sandbox.paypal.com/nvp?VERSION=56.0&USER=vendeur_TOTO&P
WD=SEFYZAZAZHN1S&SIGNATURE=Fgfgf8q5Tuj64n&METHOD=SetExpres
sCheckout&AMT=10.00&RETURNURL=http://www.monsite.com/retur
n.php&CANCELURL=http://www.monsite.com/cancel.php.
```

Paypal

- L’API Paypal retournera le token :
  - **TOKEN**=EC-28P55904SP346364Y ;
  - **TIMESTAMP**=2009-03-10T11:43:51Z ;
  - **CORRELATIONID**=62da379c625d6 (jeton de déboguage) ;
  - **ACK**=Success ;
  - **VERSION**=56.0 (version de l'API utilisée) ;
  - **BUILD**=854529

Paypal

GetExpressCheckoutDetails : permet de vérifier la transaction

Redirigez le client sur le site Paypal en y associant le token a la fonction :

https://www.sandbox.paypal.com/webscr&cmd=_express-checkout&token=EC-28P55904SP346364Y

https://cms.paypal.com/fr/cgi-bin/?&cmd=_render-content&content_ID=developer/e_howto_api_nvp_r_GetExpressCheckoutDetails

Paypal

L'API GetExpressCheckoutDetails permettra de récupérer toutes les informations concernants le paiement :

[TOKEN] => EC-28P55904SP346364Y
[TIMESTAMP] => 2009-06-08T13:01:37Z
[CORRELATIONID] => a654hfzc8bj5f
[ACK] => Success
[VERSION] => 56.0
[BUILD] => 942987
[EMAIL] => acheteur1243025979_per@siteduzero.com
[PAYERID] => G5ZMU94EQSKJS
[PAYERSTATUS] => unverified
[FIRSTNAME] => Test
[LASTNAME] => User
[COUNTRYCODE] => FR
[SHIPTONAME] => Test User
[SHIPTOSTREET] => 1 rue des fleurs
[SHIPTOCITY] => Strasbourg
[SHIPTOSTATE] => Alsace
[SHIPTOZIP] => 67000
[SHIPTOCOUNTRYCODE] => FR
[SHIPTOCOUNTRYNAME] => France
[ADDRESSSTATUS] => Unconfirmed
[CURRENCYCODE] => EUR
[AMT] => 10.00
[SHIPPINGAMT] => 0.00
[HANDLINGAMT] => 0.00
[TAXAMT] => 0.00
[INSURANCEAMT] => 0.00
[SHIPDISCAMT] => 0.00

Paypal

DoExpressCheckoutPayment : permet de valider le paiement :

Communiquer plusieurs paramètres : USER, PWD, VERSION, SIGNATURE, TOKEN) Ainsi que les paramètres suivants :

- **PAYMENTACTION** a la valeur **sale** (vente finale)
- **PAYERID** reprend l'identifiant du paiement attribué par PayPal. Cet identifiant est unique, il a été transmis a **RETURNURL**
- **AMT**, reprend le montant défini lors de l'appel de la méthode **SetExpressCheckout**.
- **CURRENCYCODE** défini la devise le montant

https://developer.paypal.com/docs/classic/api/merchant/DoExpressCheckoutPayment_API_Operation_NVP/

RESTFUL Paypal

- Paypal propose une API Restful :

https://developer.paypal.com/docs/api/

- Tuto :

https://developer.paypal.com/docs/integration/direct/make-your-first-call/