Corrigé méthodologique

Ce corrigé a pour objectif de vous permettre de vous améliorer, en explicitant les principales erreurs rencontrées et en proposant des réponses détaillées.

Question 3 (15 points) – synthèse

Un client vous contacte car il souhaite répondre à un appel d'offre du ministère de l'éducation nationale visant à diffuser en ligne des versions numérisées de différents manuels scolaires, archivés au sein d'une base de donnée Oracle.

Ce client dispose déjà d'un outil de publication de ressources documentaires sur le web écrit principalement en PHP, et souhaiterait l'adapter afin de répondre à l'appel d'offre.

Cet outil utilise une base de donnée mySQL. Il dispose également d'une librairie de rendu en HTML.

Le client vous demande de lui indiquer si il pourra proposer son outil pour répondre à l'appel d'offre

Expliquez comment vous procéderez afin de lui donner une réponse rapide et complète.

Question 4 (8 points)

Soit le code HTML suivant donné en fin de sujet.

1. Indiquez ce qui ne respecte pas la norme et pourquoi.

2. Indiquez ce qui va à l'encontre des bonnes pratiques et pourquoi.

3. Indiquez ce qui ne respecte pas les critères d'accessibilité et pourquoi.

Chaque problème détecté rapporte 0,5 point ; chaque problème correctement classé rapporte 0,5 points supplémentaires.

Il est possible de proposer un tableau pour répondre (voir tableau 1 page 2).

Il y avait en tout 14 problèmes à détecter, soit 7 points possibles, et 7points supplémentaires pour le classement (soit 14 points possibles en tout). La note maximale en cas de dépassement a été ramenée à 8 points pour cette question.

Question 6 (3 points)

Expliquez les différentes étapes exécutées par le serveur Web lorsqu'il reçoit une requête pour une page ayant une extension .php. Vous envisagerez les principaux cas possibles.

Il faut expliquer ce que fait le SERVEUR, pas le client ! Dans les grandes lignes :

1. résolution de l'a requête par le serveur : redirection via .htaccess ou directive éventuelle ;
2. vérification de l'existence de la ressource : erreur 404 sinon
3. vérification des permissions : erreur 403 sinon

4. transmission du fichier à l'interpréteur PHP (module d'apache) si ce module existe, sinon, affichage du fichier php comme texte brut. Les variables du serveurs sont également rendues accessibles (GET, POST, SESSION, etc.)
5. si module php, analyse du fichier ; déclenchement de l'interpréteur à chaque instruction <? php ….?> ; utilisation des variables CGI (GET, POST, etc...).
6. éventuellement, connexion à d'autres serveurs (base de données par exemple)
7. production d'un document de sortie par l'interpréteur php (la plupart du temps, du html, mais texte, image, xml... possibles également!).
8. envoie de la réponse au client sous la forme d'un fichier. Des messages d'erreurs peuvent éventuellement être inscrits dans ce fichier.

	ligne(s)	**Problème**	**Explication**
erreurs de norme	1	pas de doctype	un doctype est obligatoire <!doctype html> pour html5 par exemple
	1	l'attribut xmlns a une valeur erronée pour un document HTML5	
	4 et 5	les balises link sont fermées par des balises style	la balise link devrait être auto-fermante
	13	la balise tr n'est jamais fermée	c'est toléré en html5 pour certaines balises mais pas celle-là !
bonnes pratiques non respectées	9	balise meta en majuscule	toute balise doit écrite en minuscule, c'est même parfois une erreur de norme
	13	indentation inline	il vaut mieux indenter systématiquement le code pour une meilleure lisibilité.
	15	class="content" plutôt que id="content"	utiliser des id plutôt que des classes pour tout élément de structuration unique sur une page
	19	emploi de br	préférer l'utilisation d'une structure (p par exemple) combinée à des propriétés CSS pour gérer les espacements dans une page
	18	texte hors balise	structurer le document en encadrant tout contenu par des balises (ici l'usage d'un paragraphe dispenserait du br)
accessibilité	1	pas de langue déclarée	ajouter lang="fr" par exemple
	13	utilisation de tableaux pour faire de la mise en forme	remplacer par une liste, encapsulée dans une division menu ou nav
	13	pas de description de la table	caption, title, summary...
	16	titres h1 et h2 mal ordonnés	
		pas d'éléments facilitant la navigation	accesskeys, tabindex, etc.

Tableau 1: Réponses à la question 4 (classées par type d'erreurs)

Question 7 (3 points)

Pourquoi est-il préférable d'utiliser PDO plutôt que la librairie mysql en PHP ?

Illustrez votre réponse avec un exemple technique.

On attend une réponse à la question **ET** un exemple technique pour illustrer.

Réponse :

PDO est une librairie PHP, intégrée dans les distributions habituelles de PHP, permettant d'abstraire les mécanismes de gestion et d'utilisation de bases de données des SGBD comme mysql, postgres ou oracle. La librairie mysql ne permet de travailler qu'avec le SGBD mysql. Le code ainsi produit ne sera donc pas réutilisable immédiatement si on change de SGBD.

De plus, PDO est une librairie orientée objet, elle permet de gérer les exceptions, et dispose également d'un mécanisme de préparation de requête permettant de limiter les failles de sécurité comme les injections.

Ainsi, si on souhaite paramétrer le SGBD utilisé dans une application PHP, il suffit de réaliser une fonction de connexion qu'il faudra juste modifier lorsqu'on change de SGBD, ou encore mieux, utiliser un fichier de paramétrage et utiliser un paramètre pour le nom du SGBD dans la fonction de connexion.

Il était possible de donner également le code, de montrer un mécanisme de gestion d'exception ou de préparation de requête...

Question 8 (3 points)

Expliquez ce qu'est une feuille de style responsive et comment la mettre en place.

On ne demande pas ce qu'est une feuille CSS !

Réponse possible

Une feuille de style responsive est une feuille de style CSS permettant de garantir l'affichage d'une page sur tout type de dispositif, quelque soit la résolution et la taille d'affichage, ou le type de média (imprimante, écran, etc.).

Pour la mettre en place, il existe plusieurs techniques que l'on peut combiner :

1. utiliser des unités proportionnelles dans la feuille CSS (em, %, vh, etc.) ;
2. utiliser des feuilles de styles alternatives ;
3. depuis CSS3, utiliser des media queries, c'est à dire des directives ne s'exécutant que si la condition donnée est vérifiée. Ces directives peuvent porter sur les dimensions ou le type de media.

L'utilisation des styles responsive contribue notamment à la maintenabilité et la portabilité d'un site web, ainsi qu'à son accessibilité.

Question 10 (3 points)

Expliquez les différences conceptuelles d'approche entre REST et WSDL

On attend une réponse sur les différences de concepts entre REST et WSDL, pas une copie du cours.

Il est possible de répondre en deux temps : différences conceptuelles sur le fond, et différences conceptuelles sur la forme.

Réponse :

REST et WSDL sont deux manières différentes d'appréhender un WebService.

Tout d'abord, REST est un principe de fonctionnement d'un WebService, tandis que WSDL est un format de document XML décrivant un WebService.

REST utilise l'architecture RestFull : une URL permet d'accéder aux différentes parties du WebService. On peut parler d'approche implicite, puisque REST ne propose pas de description à proprement parler du webservice. Cette approche considère l'URL même comme étant auto-explicative (par exemple : http://www.site.com/ws/calcul/somme/5/3 peut représenter une ressource donnant le résultat de la somme de 5 et 3).

WSDL est un langage de description complète d'un webservice (données échangées, communication, types, etc.). Il encapsule un ou plusieurs sous-langage de description comme SOAP par exemple. On peut parler d'approche explicite, puisque tout le fonctionnement du WebService est exprimé formellement.

REST vient historiquement des approches web, donc connoté ressource/service.

WSDL vient historiquement des approches logicielles, donc connoté client/serveur.

Question 15 (5 points)

Que fait la fonction xsl donnée? Donnez un exemple d'utilisation pour le document fourni.

La question comprend deux parties :

Que fait la fonction *xsl donnée?*

Attention à ne pas confondre fonction et feuille XSL !

```
<xsl:function name="dtb:mystere" as="xs:boolean?">
  <xsl:param name="arg1" as="xs:boolean?"/>
  <xsl:param name="arg2" as="xs:boolean?"/>

  <xsl:sequence select="$arg1 != $arg2"/>

</xsl:function>
```

Réponse :

La fonction dtb:mystere prend deux paramètres booléens optionnels et renvoie un booléen.

La valeur de retour est vrai si les deux paramètres sont différents, faux si ils sont égaux (il s'agit d'un ou exclusif).

Donnez un exemple d'utilisation pour le document fourni.

Il faut proposer un exemple d'utilisation d'une fonction prenant deux booléens en paramètre et réalisant un ou exclusif.

Par exemple, avec le document fourni, afficher un paragraphe (balise p) uniquement si il possède soit un id, soit une classe :

```
<xsl:template match="p">
        <xsl:if test="dtb:mystere(@class, @id)">
                <xsl:value-of select="."/>
        </xsl:if>
</xsl:template>
```

Question 16 (1 point)

Donnez trois différences entre DTD et XSD. Inutile de détailler.

La réponse à cette question a été donnée au moins 5 fois en cours et en TP. Il a été signalé à pratiquement chaque séance que cette question était systématiquement posée en DS.

Question 17 (4 points)

Donnez un exemple d'utilisation préférable de SAX par rapport à DOM

Donnez un exemple d'utilisation préférable de DOM par rapport à SAX

On attend deux **exemples justifiés**, pas une comparaison de SAX et DOM !

Réponse :

SAX et DOM sont deux API permettant de représenter un document XML.

SAX a une approche événementielle, donc conviendra particulièrement à tout processus de traitement événementiel de l'information. Par exemple, soit un document XML contenant une liste non ordonnée d'information : on souhaite rechercher et extraire une information particulière de ce document. SAX permettra d'arréter le traitement dès que l'information est trouvée.

DOM a une approche arborescente du document XML : il doit avant tout charger intégralement en mémoire le document pour pouvoir l'exploiter. Si on cherche à vérifier la cohérence d'un document avant de l'exploiter, c'est une bonne solution. Par exemple, on dispose d'un fichier xhtml dont on veut vérifier la syntaxe avant de l'exploiter : un chargement avec DOM permettra d'interrompre les traitements si le fichier n'est pas syntaxiquement correct et/ou contient des erreurs.

D'autres exemples justifiés seraient tout aussi pertinents !

Question 18 (16 points) - Synthèse

Un libraire souhaite diffuser son catalogue en ligne et offrir la possibilité à d'autres sites d'acheter ses livres. Il souhaite également publier son catalogue sous forme de pages html (une page par livre).

Il a identifié plusieurs opérations qu'il souhaite exposer : acheter(numero du livre, quantite), retourner(numero de la commande), réserver(numéro du livre, quantité).

Il n'est pas possible de réserver plus de 10 exemplaires du même livre.

Il dispose déjà d'un fichier xml contenant l'ensemble des informations sur ses livres, et d'un logiciel installé sur son ordinateur pour gérer ce catalogue (mise à jour, ajout, suppression, etc.).

Proposez–lui une solution. Vous détaillerez techniquement les différentes configurations que vous allez mettre en place, et indiquerez éventuellement un pseudo code.

Il s'agit d'une question de synthèse ! Il faut prendre le temps d'analyser tout le problème.

1) lecture rapide

Un libraire souhaite diffuser son catalogue en ligne

Le client attend une solution web.

et offrir la possibilité à d'autres sites d'acheter ses livres.

Les fonctionnalités doivent être accessibles à d'autres sites (pas des clients finaux comme dans le cas d'un site web indépendant). On pressent le web service....

Il souhaite également publier son catalogue sous forme de pages html (une page par livre).

Il y a un **existant** ! Il faut tout de suite se demander si cet existant sera utilisable dans la solution que nous allons proposer ; il faudra donc l'évaluer.

Il a identifié plusieurs opérations qu'il souhaite exposer :

La formulation doit vous faire **immédiatement** vous faire penser à un web service !!!

acheter(numero du livre, quantite), retourner(numero de la commande), réserver(numéro du livre, quantité)

Voici les principales opérations du web service, mais vous devez vous demander si il manque quelquechose : par exemple, **comment identifie-t-on un acheteur** ? Rien n'est dit dans le sujet à ce propos, il va falloir cependant que vous preniez en compte cette contrainte cachée !

Il n'est pas possible de réserver plus de 10 exemplaires du même livre.

C'est une simple contrainte à intégrer lors de l'opération de réservation.

Il dispose déjà d'un fichier xml contenant l'ensemble des informations sur ses livres,

Voilà le format de l'existant ! Il est dit que l'ensemble des informations est contenu dans ce fichier, mais peut-être pas tout ce dont nous allons avoir besoin par la suite (commande, réservation par

exemple) ? Il va falloir étudier ce format et **imaginer les différents cas possibles**.

et d'un logiciel installé sur son ordinateur pour gérer ce catalogue (mise à jour, ajout, suppression, etc.).

L'existant comprend donc AUSSI un logiciel de gestion du catalogue, il va falloir étudier ce logiciel et dans la mesure du possible le conserver (ne pas réinventer la roue!). Le logiciel fonctionne en local, pas en réseau, il va donc falloir mettre en place un système de synchronisation entre le fichier généré par le logiciel et le fichier utilisé dans la solution. Le logiciel peut peut-être se configurer en ce sens, ou être modifié si il est open-source, etc.

Proposez–lui une solution.

Nous avons vu que cette solution est conditionnée par de nombreux paramètres : il va falloir prévoir en réalité les différents cas possibles

Vous détaillerez techniquement les différentes configurations que vous allez mettre en place,

Il s'agit bien d'imaginer l'architecture complète de la solution !

et indiquerez éventuellement un pseudo code.

2) Analyse :

existant :

- un fichier de données en XML représentant les livres

- un logiciel permettant de manipuler les informations sur ces livres.

demandes reformulées :

- mise en place d'une solution web : serveur http au minimum, probablement php, peut-être base de données

- mise en place d'un web service

- génération des pages HTML à partir du fichier xml des livres, synchroniser ce fichier avec le logiciel et le web.

- trouver comment gérer les clients et les commandes (le format xml le permet-il?)

3) Proposition de réponse pour la question

Nous disposons d'un fichier xml géré par un logiciel permettant de manipuler le catalogue du libraire. Il faudrait étudier en détail ce fichier, afin de savoir si il contient également les structures nécessaires permettant de gérer une commande et des clients. De plus, ce fichier est un fichier local, ce qui va poser des problèmes de synchronisation entre les données du libraire sur son ordinateur et les données en lignes.

Dans l'hypothèse où ce fichier convient, une solution pour le synchroniser serait de mettre en place un script rechargeant régulièrement ce fichier et gérant les transactions locales et distantes. Dans l'idéal, inclure ce script dans le logiciel du libraire-même si celui-ci est ouvert, sinon, un programme utilisant les API SAX ou DOM, installé en complément sur l'ordinateur du libraire, pourrait suffire.

Dans l'hypothèse où le fichier ne convient pas (ne contient pas toutes les informations), il faudra ajouter une solution permettant de représenter un client et une commande : soit une BD, soit un autre fichier XML. Quoiqu'il en soit, la solution devra prendre en compte les accès concurrents au fichier XML ou à la BD (plusieurs transactions sont possibles en parallèle).

La production des pages HTML peut être réalisée en exécutant un script php manipulant une transformation XSL sur le fichier du libraire, de manière dynamique. Cela permet de préserver les habitudes du libraire qui pourra continuer à utiliser son logiciel, la synchronisation se faisant via un script local. Le libraire utilisera son logiciel et mettre de manière transparente à jour les informations sur son site.

Il faudra s'assurer que les opérations inverses sont également possibles : une commande sur le site devra pouvoir mettre à jour le fichier XML et répercuter l'information dans le logiciel du libraire.

Si ces conditions ne sont pas réunies, il faudra expliquer au libraire que ses habitudes et usages ne sont pas compatibles avec ce qu'il souhaite, et lui proposer une solution 100 % web. L'initialisation des données pourra quand même se faire avec un script xsl et le fichier xml.

La deuxième partie du problème concerne la mise à disposition des procédures d'achat, de réservation et de retour de commandes à d'autres sites. La manière la plus simple de procéder consiste à mettre en place un web service pour ces opérations. Nous utiliserons ici une solution basée sur l'approche REST car les opérations sont relativement simples.

Il faudra cependant prendre en compte l'identité du client, chose qui n'est pas prévu par le libraire.

L'architecture REST pourrait avoir la structuration suivante :

http://adresse_du_ws/OPERATION/PARAM1/PARAM2/...

acheter : http://adresse_du_ws/acheter/client/livre/quantite/ donnera un numéro de commande

réserver : http://adresse_du_ws/reserver/client/livre/quantite/ en vérifiant que quantite<10

retourner : http://adresse_du_ws/retour/client/commande/

Il faudra vérifier : que le client existe, l'authentifier, que la commande existe pour le client, etc.

Enfin, la mise en place de cette solution nécessitera a minima l'installation d'un serveur web, d'un module pour PHP, et éventuellement d'une base de données si le client abandonne son logiciel ou préfère gérer ses clients et commandes dans une solution 100 % web.

Sur le même principe, vous pourriez tout à fait proposer d'autres solutions pertinentes si vous les justifiez (wsdl, mise en place de format xml intermédiaire complétant les manques possibles du fichier d'origine, etc.). Il est également possible de proposer un schéma présentant votre solution.