

Groupe de travail Réseau
Request for Comments : 3648
 Catégorie : En cours de normalisation
 Traduction Claude Brière de L'Isle

J. Whitehead, U.C. Santa Cruz
 J. Reschke, éd., greenbytes
 décembre 2003

Protocole des collections ordonnées des auteurs et des versions distribuées sur la Toile (WebDAV)

Le présent document spécifie un protocole Internet en cours de normalisation pour la communauté de l'Internet, et appelle à des discussions et des suggestions pour son amélioration. Prière de se reporter à l'édition actuelle du STD 1 "Normes des protocoles officiels de l'Internet" pour connaître l'état de normalisation et le statut de ce protocole. La distribution du présent mémoire n'est soumise à aucune restriction.

Notice de copyright

Copyright (C) The Internet Society (2004).

Résumé

La présente spécification étend le protocole de collecte des auteurs et des versions distribuée sur la Toile (WebDAV, *Web Distributed Authoring and Versioning*) pour prendre en charge l'ordre côté serveur de membres d'une collection. Les ordres qui ne sont pas fondés sur des valeurs de propriété sont d'un intérêt particulier, car ils ne peuvent pas être réalisés en utilisant l'option de rangement d'un protocole de recherche et ne peuvent pas être maintenus automatiquement par le serveur. Les éléments de protocole sont définis pour laisser les clients spécifier la position dans l'ordre de chaque membre de la collection, ainsi que la sémantique qui gouverne l'ordre.

Table des Matières

1. Conventions de notation.....	2
2. Introduction.....	2
3. Terminologie.....	3
4. Vue d'ensemble des collections ordonnées.....	3
4.1 Propriétés de collection supplémentaires.....	3
5. Création d'une collection ordonnée.....	4
5.1 Généralités.....	4
5.2 Exemple : Création d'une collection ordonnée.....	4
6. Réglage de la position d'un membre de collection.....	5
6.1 Généralités.....	5
6.2 Exemples : réglage de la position d'un membre de collection.....	6
6.3 Exemples : changement de dénomination d'un membre d'une collection ordonnées.....	6
7. Changement de l'ordre d'une collection : méthode ORDERPATCH.....	6
7.1 Exemple : Changement de l'ordre d'une collection.....	8
7.2 Exemple : échec d'une demande ORDERPATCH.....	8
8. Liste des membres d'une collection ordonnée.....	9
8.1 Exemple : PROPFIND sur une collection ordonnée.....	10
9. Relations avec les versions de collections.....	12
9.1 Propriétés des versions de collection.....	12
9.2 Sémantique supplémentaire de CHECKIN.....	12
9.3 Sémantique supplémentaire de CHECKOUT.....	12
9.4 Sémantique supplémentaire de UNCHECKOUT, UPDATE, et MERGE.....	13
10. Document de découverte de capacité.....	13
10.1 Exemple : Utilisation d'OPTIONS pour la découverte de la prise en charge de l'ordre.....	13
10.2 Exemple : Utilisation de propriétés vives pour la découverte de l'ordre.....	13
11. Considérations sur la sécurité.....	14
11.1 Dénier de service et DAV:ordering-type.....	15
12. Considérations d'internationalisation.....	15
13. Considérations relatives à l'IANA.....	15
14. Propriété intellectuelle.....	15
15. Contributeurs.....	15
16. Remerciements.....	16
17. Références normatives.....	16
Appendice A. Extensions à la définition de type de document WebDAV.....	16

Index.....	16
Adresse des auteurs.....	17
Déclaration complète de droits de reproduction.....	17

1. Conventions de notation

Comme le présent document décrit un ensemble d'extensions au protocole de collecte des auteurs et des versions distribuée sur la Toile [RFC2518], qui est lui-même une extension du protocole HTTP/1.1, le BNF augmenté utilisé ici pour décrire les éléments de protocole est exactement le même que celui décrit au paragraphe 2.1 de HTTP [RFC2616]. Comme ce BNF augmenté utilise les règles de production de base fournies au paragraphe 2.2 de HTTP, ces règles s'appliquent aussi au présent document.

Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT", et "FACULTATIF" en majuscules dans ce document sont à interpréter comme décrit dans le BCP 14, [RFC2119].

Le présent document utilise des fragments de DTD XML comme une pure convention de notation. Les corps de demande et de réponse WebDAV ne peuvent pas être validés à cause des règles d'extensibilité spécifiques définies à la Section 23 de la [RFC2518] et au fait que tous les éléments XML définis par la présente spécification utilisent le nom d'espace de nom XML "DAV:". En particulier :

1. les noms d'élément utilisent l'espace de noms "DAV:",
2. l'ordre des éléments n'est pas pertinent,
3. les éléments d'extension (éléments non encore définis comme éléments fils valides) peuvent être ajoutés partout, sauf mention contraire explicite,
4. les attributs d'extension (attributs non encore définis comme valides pour cet élément) peuvent être ajoutés partout, sauf mention contraire explicite.

2. Introduction

La présente spécification s'établit sur l'infrastructure de collection fournie par le protocole de collecte des informations d'auteurs répartie de WebDAV, en ajoutant la prise en charge du rangement des membres de la collection du côté du serveur.

Il y a de nombreux scénarios dans lesquels il est utile d'imposer un ordre à une collection chez le serveur, afin d'exprimer un ordre d'accès recommandé, ou un ordre de l'historique des révisions. Les membres d'une collection peuvent représenter les pages d'un livre, qui doivent être présentées dans l'ordre si on veut qu'elles aient un sens, ou un instructeur peut créer une collection de cours qu'il veut voir affichés dans l'ordre dans lequel ils doivent être lus.

Les rangements peuvent se fonder sur des valeurs de propriétés, mais ce n'est pas toujours le cas. Les ressources dans la collection peuvent ne pas avoir de propriétés qui puissent être utilisées pour prendre en charge l'ordre désiré. Les ordres fondés sur des propriétés peuvent être obtenus en utilisant l'option de rangement d'un protocole de recherche, mais les ordres qui ne sont pas fondés sur des propriétés ne le peuvent pas. Ces ordres doivent généralement être maintenus par une personne.

Le protocole de rangement défini ici se focalise sur la prise en charge de tels ordres maintenus par l'homme. Ses éléments de protocole permettent aux clients de spécifier la position de chaque membre de la collection dans l'ordre de la collection, ainsi que la sémantique qui gouverne l'ordre. Le protocole est conçu pour permettre à l'avenir une prise en charge supplémentaire d'ordres dont le maintien est fait automatiquement par le serveur.

Le reste du présent document est structuré comme suit : la Section 3 définit la terminologie qui sera utilisée tout au long de la spécification. La Section 4 donne une vue d'ensemble des collections ordonnées. La Section 5 décrit comment créer une collection ordonnée, et la Section 6 discute comment établir la position d'un membre dans le rangement d'une collection. La Section 7 explique comment changer l'ordre d'une collection. La Section 8 discute des listes de membres d'une collection ordonnée. La Section 9 discute de l'impact sur les collections à version contrôlée (comme défini dans la [RFC3253]). La Section 10 décrit la découverte des capacités. Les Sections 11 à 13 discutent des considérations de sécurité, d'internationalisation, et de l'IANA. Les sections restantes apportent les informations de soutien.

3. Terminologie

La terminologie utilisée suit celle des [RFC2518] et [RFC3253]. La définition des termes "ressource", "identifiant de ressource universel" (URI, *Uniform Resource Identifier*), et "localisateur de ressource universel" (URL, *Uniform Resource Locator*) est fournie dans la [RFC2396].

Collection ordonnée : collection pour laquelle il est garanti que les résultats d'une demande PROPFIND sont dans l'ordre spécifié pour cette collection.

Collection non ordonnée : collection pour laquelle le client ne peut pas compter sur la répétabilité de l'ordre des résultats d'une demande PROPFIND.

Ordre maintenu par le client : ordre des membres d'une collection qui est maintenu sur le serveur sur la base des demandes du client qui spécifient la position de chaque membre de la collection dans cet ordre.

Ordre maintenu par le serveur : ordre des membres d'une collection qui est maintenu automatiquement par le serveur, sur la base d'un choix du client de la sémantique de rangement.

Sémantique de rangement : en général, la sémantique de rangement est l'ensemble des structures ou significations appliquées à l'ordre des membres d'une certaine collection. Dans le présent document, "sémantique de rangement" se réfère précisément à la structure spécifiée dans la propriété DAV:ordering-type. Voir au paragraphe 4.1.1 plus d'informations sur DAV:ordering-type.

Le présent document utilise les termes "précondition", "postcondition" et "propriété protégée" comme défini dans la [RFC3253]. Les serveurs DOIVENT faire rapport des échecs de pré/postcondition comme décrit au paragraphe 1.6 du présent document.

4. Vue d'ensemble des collections ordonnées

Si une collection n'est pas ordonnée, le client ne peut pas compter sur la répétabilité de l'ordre des résultats d'une demande PROPFIND. En spécifiant un ordre pour une collection, un client demande au serveur de suivre cet ordre chaque fois qu'il répond à une demande PROPFIND sur cette collection.

Les ordres du côté du serveur peuvent être entretenus par le client ou entretenus par le serveur. Pour les ordres entretenus par le client, un client doit spécifier la position dans l'ordre de chacun des membres de la collection, soit lorsque le membre est ajouté à la collection (en utilisant l'en-tête Position (Section 6)) soit ultérieurement (en utilisant la méthode ORDERPATCH (Section 7)). Pour les ordres entretenus par le serveur, le serveur positionne automatiquement chacun des membres de la collection selon la sémantique de l'ordre. La présente spécification ne prend en charge que les ordres entretenus par le client, mais elle est conçue pour permettre une future extension avec des ordres entretenus par le serveur.

Une collection qui prend en charge l'ordre n'est pas obligée d'être ordonnée. Si une collection est ordonnée, chacun des URI de ses membres internes DOIT apparaître exactement une fois, et l'ordre NE DOIT PAS inclure d'URI qui ne soit pas de membre interne de la collection. Le serveur est responsable de la mise en application de ces contraintes sur l'ordre. Le serveur DOIT retirer de l'ordre un URI de membre interne lorsque celui-ci est retiré de la collection. Retirer un membre interne NE DOIT PAS affecter l'ordre des membres internes restants. Le serveur DOIT ajouter à l'ordre un URI de membre interne lorsque celui-ci est ajouté à la collection.

Un seul ordre peut être attaché à une collection. Plusieurs ordres de la même ressource peuvent être réalisés en créant plusieurs collections qui se réfèrent à cette ressource, et en attachant un ordre différent à chaque collection.

Un ordre est considéré faire partie de l'état d'une ressource de collection. Par conséquent, l'ordre est le même quel que soit l'URI utilisé pour accéder à la collection et il est protégé par des verrous ou des contraintes de contrôle d'accès sur la collection.

4.1 Propriétés de collection supplémentaires

Une demande PROPFIND DAV:allprop NE DEVRAIT PAS retourner une des propriétés définies dans le présent document.

4.1.1 DAV:ordering-type (protégé)

La propriété DAV:ordering-type indique si la collection est ordonnée et, si elle l'est, identifie de façon univoque la sémantique de rangement. Elle peut aussi pointer sur une explication de la sémantique sous une forme lisible par l'homme et/ou la machine. Au minimum, cela permet aux personnes qui ajoutent des membres à la collection de comprendre où les positionner dans l'ordre. Cette propriété ne peut pas être réglée en utilisant PROPPATCH. Sa valeur peut seulement être réglée en incluant l'en-tête Ordering-Type avec une demande MKCOL ou en soumettant une demande ORDERPATCH.

Les types d'ordres sont identifiés par les URI qui identifient de façon univoque la sémantique de rangement de la collection. Les deux URI suivants sont prédéfinis :

DAV:custom : cette valeur indique que la collection est ordonnée, mais la sémantique qui gouverne l'ordre n'est pas annoncée.

DAV:unordered : cette valeur indique que la collection n'est pas ordonnée. C'est à dire que le client ne peut pas compter sur la répétabilité de l'ordre des résultats d'une demande PROPFIND.

Un client capable d'ordre qui interagit avec un serveur sans capacité d'ordre (par exemple, qui est mis en œuvre seulement selon la [RFC2518]) DEVRAIT supposer que la collection est sans ordre si une collection n'a pas la propriété DAV:ordering-type.

<!ELEMENT ordering-type (href) >

5. Création d'une collection ordonnée

5.1 Généralités

Lorsque une collection est créée, le client PEUT demander qu'elle soit ordonnée et spécifier la sémantique de rangement en utilisant le nouvel en-tête Ordering-Type (défini plus loin) avec une demande MKCOL.

Pour les collections qui sont ordonnées, le client DEVRAIT identifier la sémantique de rangement avec un URI dans l'en-tête Ordering-Type, bien que le client PUISSE simplement régler la valeur de l'en-tête à DAV:custom pour indiquer que la collection est ordonnée mais que la sémantique de rangement n'est pas annoncée. Régler la valeur d'un URI qui identifie la sémantique de rangement fournit des informations qu'un utilisateur humain ou un paquetage logiciel va utiliser pour insérer intelligemment de nouveaux membres d'une collection dans l'ordre. Bien que l'URI dans l'en-tête Ordering-Type PUISSE pointer sur une ressource qui contient une définition de la sémantique de rangement, les clients NE DEVRAIENT PAS accéder à cette ressource pour éviter de surcharger leur serveur. Une valeur de DAV:unordered dans l'en-tête Ordering-Type indique que le client veut que la collection soit sans ordre. Si l'en-tête Ordering-Type n'est pas présent, la collection sera sans ordre.

Directives de comportement supplémentaires :

Ordering-Type = "Ordering-Type" ":" absoluteURI ; URI absolu : voir la section 3 de la RFC2396.

L'URI "DAV:unordered" indique que la collection n'est pas ordonnée, tandis que "DAV:custom" indique que la collection est à ordonner, mais la sémantique de rangement n'est pas annoncée. Toute autre valeur d'URI indique que la collection est ordonnée, et identifie la sémantique de rangement.

Préconditions supplémentaires :

(DAV:ordered-collections-supported) : le serveur DOIT prendre en charge les collections ordonnées dans la partie de l'espace de noms d'URL identifiée par l'URL de demande.

Postconditions supplémentaires :

(DAV:ordering-type-set) : si l'en-tête Ordering-Type était présent, la demande DOIT avoir créé une nouvelle ressource de collection dont le DAV:ordering-type a été réglé conformément à l'en-tête Ordering-Type de demande. La collection DOIT être ordonnée sauf si le type d'ordre est "DAV:unordered".

5.2. Exemple : Création d'une collection ordonnée

>> Demande :

```
MKCOL /theNorth/ HTTP/1.1
Host: example.org
```

Ordering-Type: http://example.org/orderings/compass.html

>> Réponse :

HTTP/1.1 201 Created

Dans cet exemple, une nouvelle collection ordonnée a été créée. Sa propriété DAV:ordering-type a l'URI provenant de l'en-tête Ordering-Type qui a la valeur http://example.org/orderings/compass.html. Dans ce cas, l'URI identifie la sémantique qui gouverne un ordre entretenu par le client. Comme de nouveaux membres sont ajoutés à la collection, les clients ou les utilisateurs finaux peuvent utiliser la sémantique pour déterminer où positionner les nouveaux membres dans l'ordre.

6. Réglage de la position d'un membre de collection

6.1 Généralités

Lorsque un nouveau membre est ajouté à une collection avec un ordre entretenu par un client (par exemple, avec PUT, COPY, ou MKCOL) sa position dans l'ordre peut être réglée avec le nouvel en-tête Position. L'en-tête Position permet au client de spécifier qu'un URI de membre interne devrait être en premier dans l'ordre de la collection, en dernier dans l'ordre de la collection, immédiatement avant l'URI de quelque autre membre interne dans l'ordre de la collection, ou immédiatement après l'URI d'un autre membre interne dans l'ordre de la collection.

Si l'en-tête de demande Position n'est pas utilisé lors de l'ajout d'un membre à une collection ordonnée, alors :

- o si la demande est de remplacer une ressource existante, le serveur DOIT préserver l'ordre existant ;
- o si la demande est d'ajouter l'URI d'un nouveau membre interne à la collection, le serveur DOIT ajouter le nouveau membre à la fin du rangement.

Note de mise en œuvre : la présente spécification n'impose pas une mise en œuvre spécifique des opérations MOVE au sein de la même collection parente. Donc, les serveurs peuvent mettre en œuvre cela comme une simple opération de changement de nom (préservant la position du membre de la collection) ou comme une séquence de "suppressions" et "ajouts" (causant l'application de la sémantique de "ajout d'un nouveau membre"). De futures révisions de la présente spécification pourront spécifier ce comportement de façon plus précise sur la base de l'expérience.

Directives de comportement supplémentaires :

Position = "Position" ":" ("first" | "last" | ("before" | "after") segment)
; segment est défini au paragraphe 3.3 de la [RFC2396].

Le segment est interprété par rapport à la collection à laquelle est ajouté le nouveau membre.

Lorsque l'en-tête Position est présent, le serveur DOIT insérer le nouveau membre à la localisation spécifiée dans l'ordre.

Le mot-clé "first" indique que le nouveau membre est placé à la position de début dans l'ordre de la collection, tandis que "last" indique que le nouveau membre est placé à la position finale de l'ordre de la collection. Le mot clé "before" indique que le nouveau membre est ajouté à l'ordre de la collection immédiatement avant la position du membre identifié dans le segment. De même, le mot clé "after" indique que le nouveau membre est ajouté à l'ordre de la collection immédiatement à la suite de la position du membre identifié dans le segment.

Si la demande remplace une ressource existante et si l'en-tête Position est présent, le serveur DOIT supprimer l'URI de membre interne de sa position actuelle, et l'insérer à la position demandée.

Préconditions supplémentaires :

(DAV:collection-must-be-ordered) : la collection cible DOIT être ordonnée.

(DAV:segment-must-identify-member) : le segment référencé DOIT identifier une ressource qui existe et est différente de la ressource affectée.

Postconditions supplémentaires :

(DAV:position-set) : si un en-tête Position est présent, la demande DOIT créer le nouveau membre de la collection à la position spécifiée.

6.2 Exemples : réglage de la position d'un membre de collection

>> Demande :

```
COPY ~/user/dav/spec08.html HTTP/1.1
Host: example.org
Destination: http://example.org/~slein/dav/spec08.html
Position: after requirements.html
```

>> Réponse :

```
HTTP/1.1 201 Created
```

Cette demande a résulté en la création d'une nouvelle ressource à `example.org/~slein/dav/spec08.html`. L'en-tête `Position` dans cet exemple a causé l'établissement par le serveur de sa position dans l'ordre de la collection `/~slein/dav/` immédiatement après `requirements.html`.

>> Demande :

```
MOVE /i-d/draft-webdav-prot-08.txt HTTP/1.1
Host: example.org
Destination: http://example.org/~user/dav/draft-webdav-prot-08.txt
Position: first
```

>> Réponse :

```
HTTP/1.1 409 Conflict
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx
```

```
<?xml version="1.0" encoding="utf-8" ?>
<D:error xmlns:D="DAV:">
  <D:collection-must-be-ordered/>
</D:error>
```

Dans ce cas, le serveur retourne un code d'état 409 (Conflit) parce que la collection `/~user/dav/` est une collection non ordonnée. Par conséquent, le serveur est dans l'incapacité de satisfaire l'en-tête `Position`.

6.3 Exemples : changement de dénomination d'un membre d'une collection ordonnées

La séquence de demandes suivante va renommer un membre de collection tout en préservant sa position, indépendamment de la façon dont le serveur met en œuvre l'opération `MOVE` :

1. `PROPFIND` la collection à la profondeur 1, restituant la propriété `DAV:ordering-type` (un client interactif a vraisemblablement déjà fait cela afin d'afficher le contenu de la collection).
2. Si la propriété `DAV:ordering-type` est présente et n'est pas égale à `"dav:unordered"` (donc, si la collection est ordonnée) déterminer la position actuelle (comme `"first"` ou `"after x"`) et régler l'en-tête `Position` en conséquence.
3. Effectuer l'opération `MOVE`, facultativement en fournissant l'en-tête `Position` calculé à l'étape précédente.

7. Changement de l'ordre d'une collection : méthode `ORDERPATCH`

La méthode `ORDERPATCH` est utilisée pour changer la sémantique de rangement d'une collection, pour changer l'ordre des membres de la collection dans le rangement, ou les deux.

Le serveur DOIT appliquer les changements dans l'ordre où ils apparaissent dans l'élément XML d'ordre. Le serveur DOIT soit appliquer tous les changements, soit n'en appliquer aucun. Si une erreur se produit durant le traitement, tous les changements exécutés DOIVENT être annulés et un résultat d'erreur approprié être retourné.

Si une demande `ORDERPATCH` change la sémantique de rangement, mais ne spécifie pas complètement l'ordre des membres

de la collection, le serveur DOIT allouer une position dans l'ordre à chaque membre de la collection pour lequel une position n'a pas été spécifiée. Ces positions allouées par le serveur DOIVENT suivre la dernière position spécifiée par le client. Le résultat est que tous les membres pour lesquels le client a spécifié une position sont au début du rangement, suivis par tous les membres pour lesquels le serveur a alloué une position. Noter que l'ordre des positions allouées par le serveur n'est pas défini par le présent document, donc les serveurs peuvent utiliser toute règle qui semble raisonnable (par exemple, alphabétique ou par date de création).

Si une demande ORDERPATCH ne change pas la sémantique de rangement, toutes les positions de membres non spécifiées dans la demande DOIVENT rester inchangées.

Une demande de repositionnement d'un membre d'une collection à la même place dans l'ordre n'est pas une erreur.

Si une demande ORDERPATCH échoue, l'état de serveur précédant la demande DOIT être restauré.

Directives de comportement supplémentaires : Le corps de la demande DOIT être un élément DAV:orderpatch.

```
<!ELEMENT orderpatch (ordering-type?, order-member*) >
<!ELEMENT order-member (segment, position) >
<!ELEMENT position (first | last | before | after)>
<!ELEMENT segment (#PCDATA)>
<!ELEMENT first EMPTY >
<!ELEMENT last EMPTY >
<!ELEMENT before segment >
<!ELEMENT after segment >
```

Valeur de PCDATA : segment, comme défini au paragraphe 3.3 de la [RFC2396].

La propriété DAV:ordering-type est modifiée conformément à l'élément DAV:ordering-type.

L'ordre des URI de membres internes dans la collection identifiée par l'URI de demande est changé sur la base des instructions dans les éléments XML order-member. Précisément, dans l'ordre dans lequel ils apparaissent dans la demande. Les éléments XML order-member identifient les URI de membre interne dont les positions ne sont pas à changer, et décrivent leur nouvelle position dans l'ordre. Chaque nouvelle position peut être spécifiée comme première dans l'ordre, dernière dans l'ordre, immédiatement avant un autre URI de membre interne, ou immédiatement après un URI d'un autre membre interne.

Si un corps de réponse pour une demande réussie est inclus, il DOIT être un élément XML DAV:orderpatch-response. Noter que le présent document ne définit aucun élément pour le corps de réponse ORDERPATCH, mais l'élément DAV:orderpatch-response est défini pour assurer l'interopérabilité avec de futures extensions qui, elles, définiront des éléments pour le corps de réponse ORDERPATCH.

```
<!ELEMENT orderpatch-response ANY>
```

Comme plusieurs changements peuvent être demandés dans une seule demande ORDERPATCH, le serveur DOIT retourner une réponse 207 (Multi-Status) (définie dans la [RFC2518]) contenant des éléments de DAV:response soit pour l'URI de demande (lorsque le DAV:ordering-type n'a pas pu être modifié) soit pour des URI de membres de collection à repositionner (quand une demande de positionnement individuel exprimée comme un DAV:order-member n'a pas pu être satisfaite) si des problèmes se présentent.

Préconditions :

(DAV:collection-must-be-ordered) : voir au paragraphe 6.1.

(DAV:segment-must-identify-member) : voir au paragraphe 6.1.

Postconditions :

(DAV:ordering-type-set) : si le corps de la demande contenait un élément DAV:ordering-type, la demande DOIT avoir réglé la propriété DAV:ordering-type de la collection à la valeur spécifiée dans la demande.

(DAV:ordering-modified) : si le corps de la demande contenait des éléments DAV:order-member, la demande DOIT avoir réglé l'ordre des URI de membre interne dans la collection identifiés par l'URI de demande sur la base des instructions des éléments DAV:order-member.

7.1 Exemple : Changement de l'ordre d'une collection

Considérons une collection ordonnée /coll-1/, avec des liens ordonnés comme suit :

```
three.html
four.html
one.html
two.html
```

>> Demande :

```
ORDERPATCH /coll-1/ HTTP/1.1
Host: example.org
Content-Type: text/xml; charset="utf-8"
Content-Length: xxx
```

```
<?xml version="1.0" ?>
<d:orderpatch xmlns:d="DAV:">
  <d:ordering-type>
    <d:href>http://example.org/inorder.ord</d:href>
  </d:ordering-type> <d:order-member>
    <d:segment>two.html</d:segment>
    <d:position><d:first/></d:position>
  </d:order-member>
  <d:order-member>
    <d:segment>one.html</d:segment>
    <d:position><d:first/></d:position>
  </d:order-member>
  <d:order-member>
    <d:segment>three.html</d:segment>
    <d:position><d:last/></d:position>
  </d:order-member>
  <d:order-member>
    <d:segment>four.html</d:segment>
    <d:position><d:last/></d:position>
  </d:order-member>
</d:orderpatch>
```

>> Réponse :

```
HTTP/1.1 200 OK
```

Dans cet exemple, après le traitement de la demande, la sémantique de rangement de la collection est identifiée par l'URI `http://example.org/inorder.ord`. La valeur de la propriété `DAV:ordering-type` de la collection a été réglée à cet URI. La demande contient aussi des instructions pour changer les positions dans l'ordre d'URI de membres internes de la collection pour se conformer à la nouvelle sémantique de rangement. Comme des éléments `DAV:order-member` doivent être traités dans l'ordre dans lequel ils apparaissent dans la demande, `two.html` est déplacé au début du rangement, et ensuite `one.html` est déplacé au début du rangement. Puis `three.html` est déplacé à la fin du rangement, et finalement `four.html` est déplacé à la fin du rangement. Après le traitement de la demande, l'ordre de la collection est le suivant :

```
one.html
two.html
three.html
four.html
```

7.2 Exemple : échec d'une demande ORDERPATCH

Considérons une collection /coll-1/ dont les membres sont ordonnés comme suit :

```
nunavut.map
nunavut.img
baffin.map
baffin.desc
baffin.img
iqaluit.map
```

```
nunavut.desc
iqaluit.img
iqaluit.desc
```

>> Demande :

```
ORDERPATCH /coll-1/ HTTP/1.1
Host: www.nunanet.com
Content-Type: text/xml; charset="utf-8"
Content-Length: xxx
```

```
<?xml version="1.0" ?>
<d:orderpatch xmlns:d="DAV:">
  <d:order-member>
    <d:segment>nunavut.desc</d:segment>
    <d:position>
      <d:after>
        <d:segment>nunavut.map</d:segment>
      </d:after>
    </d:position>
  </d:order-member>
  <d:order-member>
    <d:segment>iqaluit.map</d:segment>
    <d:position>
      <d:after>
        <d:segment>pangnirtung.img</d:segment>
      </d:after>
    </d:position>
  </d:order-member>
</d:orderpatch>
```

>> Réponse :

```
HTTP/1.1 207 Multi-Status
Content-Type: text/xml; charset="utf-8"
Content-Length: xxx
```

```
<?xml version="1.0" ?>
<d:multistatus xmlns:d="DAV:">
  <d:response>
    <d:href>http://www.nunanet.com/coll-1/iqaluit.map</d:href>
    <d:status>HTTP/1.1 403 Forbidden</d:status>
    <d:responsedescription>
      <d:error><d:segment-must-identify-member/></d:error>
      pangnirtung.img n'est pas un membre de la collection.
    </d:responsedescription>
  </d:response>
</d:multistatus>
```

Dans cet exemple, le client a tenté de positionner iqaluit.map après un URI qui n'est pas un membre interne de la collection /coll-1/. Le serveur répond à cette erreur du client par un code d'état 403 (Interdit) indiquant l'échec de la précondition DAV:segment-must-identify-member. Parce que ORDERPATCH est une méthode atomique, la demande de repositionnement de nunavut.desc (qui aurait autrement pu réussir) a échoué aussi, mais cela n'a pas besoin d'être exprimé dans le corps de réponse multistatus.

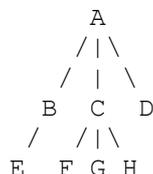
8. Liste des membres d'une collection ordonnée

Une demande PROPFIND est utilisée pour restituer une liste des membres d'une collection ordonnée, tout comme elle est utilisée pour restituer une liste des membres d'une collection non ordonnée.

Cependant, lorsque on répond à une demande PROPFIND sur une collection ordonnée, le serveur DOIT ordonner les éléments

de réponse conformément à l'ordre défini sur la collection. Si une collection n'est pas ordonnée, le client ne peut pas compter sur la répétabilité de l'ordre des résultats d'une demande PROPFIND.

Dans une réponse à une demande PROPFIND avec Depth: infinity, les membres des différentes collections peuvent être intriqués. C'est-à-dire que le serveur n'est pas obligé de faire une traversée de part en part. La seule exigence est que les membres de toute collection ordonnée apparaissent dans l'ordre défini pour cette collection. Donc, pour la hiérarchie illustrée dans la figure qui suit, où la collection A est une collection ordonnée avec l'ordre B C D,



il serait acceptable que le serveur retourne les éléments de réponse aussi bien dans l'ordre A B E C F G H D que "A B E C H G F D" (si C est non ordonnée). Dans cette réponse, B, C, et D apparaissent dans l'ordre correct, séparés par les membres d'autres collections. Les clients peuvent utiliser une série de demandes PROPFIND Depth: 1 (*de profondeur 1*) pour éviter la complexité du traitement des réponses Depth: infinity sur la base de traversées de première profondeur (*depth-first*).

8.1 Exemple : PROPFIND sur une collection ordonnée

Supposons une demande PROPFIND qui est soumise à /MyColl/, dont les membres sont ordonnés comme suit :

```

/MyColl/
 lakehazen.html
 siorapaluk.html
 iqaluit.html
 newyork.html

```

>> Demande :

```
PROPFIND /MyColl/ HTTP/1.1
```

```
Host: example.org
Depth: 1
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx
```

```
<?xml version="1.0" ?>
<D:propfind xmlns:D="DAV:">
  <D:prop xmlns:J="http://example.org/jsprops/">
    <D:ordering-type/>
    <D:resourcetype/>
    <J:latitude/>
  </D:prop>
</D:propfind>
```

>> Réponse :

```
HTTP/1.1 207 Multi-Status
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx
```

```
<?xml version="1.0" ?>
<D:multistatus xmlns:D="DAV:"
  xmlns:J="http://example.org/jsprops/">
  <D:response>
    <D:href>http://example.org/MyColl/</D:href>
    <D:propstat>
      <D:prop>
        <D:ordering-type>
```

```

    <D:href>DAV:custom</D:href>
  </D:ordering-type>
  <D:resourcetype><D:collection/></D:resourcetype>
</D:prop>
<D:status>HTTP/1.1 200 OK</D:status>
</D:propstat>
<D:propstat>
  <D:prop>
    <J:latitude/>
  </D:prop>
  <D:status>HTTP/1.1 404 Not Found</D:status>
</D:propstat>
</D:response>
<D:response>
  <D:href>http://example.org/MyColl/lakehazen.html</D:href>
  <D:propstat>
    <D:prop>
      <D:resourcetype/>
      <J:latitude>82N</J:latitude>
    </D:prop>
    <D:status>HTTP/1.1 200 OK</D:status>
  </D:propstat>
  <D:propstat>
    <D:prop>
      <D:ordering-type/>
    </D:prop>
    <D:status>HTTP/1.1 404 Not Found</D:status>
  </D:propstat>
</D:response>
<D:response>
  <D:href>
  >http://example.org/MyColl/siorapaluk.html</D:href>
  <D:propstat>
    <D:prop>
      <D:resourcetype/>
      <J:latitude>78N</J:latitude>
    </D:prop>
    <D:status>HTTP/1.1 200 OK</D:status>
  </D:propstat>
  <D:propstat>
    <D:prop>
      <D:ordering-type/>
    </D:prop>
    <D:status>HTTP/1.1 404 Not Found</D:status>
  </D:propstat>
</D:response>
<D:response>
  <D:href>http://example.org/MyColl/iqaluit.html</D:href>
  <D:propstat>
    <D:prop>
      <D:resourcetype/>
      <J:latitude>62N</J:latitude>
    </D:prop>
    <D:status>HTTP/1.1 200 OK</D:status>
  </D:propstat>
  <D:propstat>
    <D:prop>
      <D:ordering-type/>
    </D:prop>
    <D:status>HTTP/1.1 404 Not Found</D:status>
  </D:propstat>
</D:response>
<D:response>

```

```

<D:href>http://example.org/MyColl/newyork.html</D:href>
<D:propstat>
  <D:prop>
    <D:resourcetype/>
    <J:latitude>45N</J:latitude>
  </D:prop>
  <D:status>HTTP/1.1 200 OK</D:status>
</D:propstat>
<D:prop>
  <D:ordering-type/>
</D:prop>
<D:status>HTTP/1.1 404 Not Found</D:status>
</D:propstat>
</D:propstat>
</D:response>
</D:multistatus>

```

Dans cet exemple, le serveur a répondu par une liste des membres de la collection dans l'ordre défini pour la collection.

9. Relations avec les versions de collections

Les extensions de version à WebDAV [RFC3253] introduisent le concept de versions collectées, enregistrant à la fois les propriétés mortes et l'ensemble des liens internes de version contrôlée. Cette section définit comment cette caractéristique interagit avec les collections ordonnées.

La présente spécification considère à la fois le type d'ordre (propriété DAV:ordering-type) et l'ordre des membres de collection comme faisant partie de l'état d'une collection. Donc, tous deux DOIVENT être enregistrés lors des opérations CHECKIN ou VERSION-CONTROL, et tous deux DOIVENT être restaurés lors de CHECKOUT, UNCHECKOUT ou UPDATE (où, pour la compatibilité avec la RFC3253, seul l'ordre des membres de version contrôlée a besoin d'être entretenu).

9.1 Propriétés des versions de collection

9.1.1 Sémantique supplémentaire pour DAV:version-controlled-binding-set (protégé)

Pour les collections protégées, les éléments DAV:version-controlled-binding DOIVENT apparaître dans l'ordre défini pour la collection ordonnée enregistrée.

9.1.2 DAV:ordering-type (protégé)

La propriété DAV:ordering-type enregistre la propriété de DAV:ordering-type de la collection ordonnée enregistrée.

9.2 Sémantique supplémentaire de CHECKIN

Postconditions supplémentaires :

(DAV:initialize-version-controlled-bindings-ordered) : Si l'URL de demande identifiait une collection à la fois ordonnée et à version contrôlée, alors les éléments fils de DAV:version-controlled-binding-set de la nouvelle version de collection DOIVENT apparaître dans l'ordre défini pour cette collection.

(DAV:initialize-collection-version-ordering-type) : si l'URL de demande identifiait une collection à la fois ordonnée et à version contrôlée, alors la propriété DAV:ordering-type de la nouvelle version de collection DOIT être une copie de la propriété DAV:ordering-type de la collection.

9.3 Sémantique supplémentaire de CHECKOUT

Postconditions supplémentaires :

(DAV:initialize-version-history-bindings-ordered) : si la demande a été appliquée à une version de collection avec un DAV:ordering-type autre que "DAV:unordered", les liens dans la nouvelle collection de travail DOIVENT être ordonnés conformément à la propriété DAV:version-controlled-binding-set de la version de la collection.

(DAV:initialize-ordering-type) : si la demande a été appliquée à une version de collection, la propriété DAV:ordering-type de la nouvelle collection de travail DOIT être initialisée à partir de la propriété DAV:ordering-type de la version de la collection.

9.4 Sémantique supplémentaire de UNCHECKOUT, UPDATE, et MERGE

Postconditions supplémentaires :

(DAV:update-version-controlled-collection-members-ordered) : si la demande a modifié la version DAV:checked-in d'une collection à version contrôlée et si le DAV:ordering-type pour la version qui s'enregistre est non ordonné ("DAV:unordered") les membres de la version contrôlée DOIVENT être ordonnés conformément à la propriété DAV:version-controlled-binding-set de la version qui s'enregistre. L'ordre des membres d'une version non contrôlée est défini par le serveur.

(DAV:update-version-ordering-type) : si la demande a modifié la version DAV:checked-in d'une collection à version contrôlée, la propriété DAV:ordering-type DOIT être mise à jour à partir de la propriété de la version enregistrée.

10. Document de découverte de capacité

Le paragraphe 9.1 et la Section 15 de la [RFC2518] décrivent l'utilisation de classes de conformité avec l'en-tête DAV en réponse à OPTIONS, pour indiquer quelles parties des protocoles de versions d'auteurs distribuées sur la Toile la ressource prend en charge. La présente spécification définit une extension FACULTATIVE à la [RFC2518]. Elle définit une nouvelle classe de conformité, appelée "ordered-collections", à utiliser avec l'en-tête DAV dans les réponses aux demandes OPTIONS. Si une ressource de collection ne prend pas en charge l'ordre, sa réponse à une demande OPTIONS peut indiquer qu'elle le fait en mentionnant la nouvelle méthode ORDERPATCH comme une de celles qu'elle prend en charge, et en citant la nouvelle classe de conformité ordered-collections dans l'en-tête DAV.

En répondant à une demande OPTIONS, seule une collection ou une ressource nulle peut inclure des ordered-collections dans la valeur de l'en-tête DAV. En incluant des ordered-collections, la ressource indique que ses URI de membres internes peuvent être ordonnés. Cela n'a pas d'implication sur la possibilité que des collections identifiées par ses URI de membres internes soient ordonnées.

De plus, la [RFC3253] introduit les propriétés vives DAV:supported-method-set (paragraphe 3.1.3) et DAV:supported-live-property-set (paragraphe 3.1.4). Les serveurs DOIVENT prendre en charge ces propriétés comme défini dans la RFC3253.

10.1 Exemple : Utilisation d'OPTIONS pour la découverte de la prise en charge de l'ordre

>> Demande :

```
OPTIONS /somecollection/ HTTP/1.1
Host: example.org
```

>> Réponse :

```
HTTP/1.1 200 OK
Allow: OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, COPY, MOVE
Allow: MKCOL, PROPFIND, PROPPATCH, LOCK, UNLOCK, ORDERPATCH
DAV: 1, 2, ordered-collections
```

L'en-tête DAV dans la réponse indique que la ressource /somecollection/ est conforme aux niveaux 1 et 2, comme défini dans la [RFC2518]. De plus, /somecollection/ prend en charge l'ordre. L'en-tête Allow indique que les demandes ORDERPATCH peuvent être soumises à /somecollection/.

10.2 Exemple : Utilisation de propriétés vives pour la découverte de l'ordre

>> Demande :

```
PROPFIND /somecollection HTTP/1.1
Depth: 0
Content-Type: text/xml; charset="utf-8"
```

Content-Length: xxx

```
<?xml version="1.0" encoding="UTF-8" ?>
<propfind xmlns="DAV:">
  <prop>
    <supported-live-property-set/>
    <supported-method-set/>
  </prop>
</propfind>
```

>> Réponse :

HTTP/1.1 207 Multi-Status

Content-Type: text/xml; charset="utf-8"

Content-Length: xxx

```
<?xml version="1.0" encoding="utf-8" ?>
<multistatus xmlns="DAV:">
  <response>
    <href>http://example.org/somecollection</href>
    <propstat>
      <prop>
        <supported-live-property-set>
          <supported-live-property>
            <prop><ordering-type/></prop>
          </supported-live-property>
        </supported-live-property-set>
      </prop>
      <status>HTTP/1.1 200 OK</status>
    </propstat>
  </response>
</multistatus>
```

Noter que les réponses réelles DOIVENT contenir une liste complète des propriétés vives prises en charge.

11. Considérations sur la sécurité

Cette section est fournie pour informer les mises en œuvre de WebDAV des implications de ce protocole pour la sécurité.

Toutes les considérations de sécurité de HTTP/1.1 et de la spécification du protocole WebDAV d'informations d'auteur distribuées s'appliquent aussi à la présente spécification de protocole. De plus, les collections ordonnées introduisent un nouveau souci pour la sécurité. Cette question est détaillée ici.

11.1 Dénier de service et DAV:ordering-type

Il peut y avoir un risque de déni de service aux sites qui sont annoncés dans la propriété DAV:ordering-type des collections. Cependant, on prévoit que les applications largement déployées utiliseront des valeurs codées d'origine pour les sémantiques de rangement fréquemment utilisées plutôt que de rechercher la sémantique à la localisation spécifiée par DAV:ordering-type. Ce risque sera encore réduit si les clients observent la recommandation du paragraphe 5.1 que les demandes ne soient pas envoyées à l'URI dans DAV:ordering-type.

12. Considérations d'internationalisation

La présente spécification suit les pratiques de la [RFC2518] en codant tout le contenu lisible par l'homme en utilisant [XML] et dans le traitement des noms. Par conséquent, la présente spécification se conforme à la politique de jeux de caractères de l'IETF [RFC2277].

Les applications WebDAV DOIVENT prendre en charge les fonctionnalités d'étiquetage de jeu de caractères, de codage de jeu de caractères et d'étiquetage de langage de la spécification XML. Cette contrainte assure que le contenu lisible par l'homme de cette spécification se conforme à la [RFC2277].

Comme dans la [RFC2518], les noms dans la présente spécification entrent dans trois catégories : noms d'éléments de protocole tels que les méthodes et les en-têtes, noms d'éléments XML, et noms de propriétés. La dénomination des éléments de protocole suit le précédent de HTTP en utilisant des noms anglais codés en US-ASCII pour les méthodes et les en-têtes. Les noms des éléments XML utilisés dans la présente spécification sont des noms anglais codés en UTF-8.

Pour les rapports d'erreurs, la [RFC2518] suit la convention des codes d'état HTTP/1.1, incluant avec chaque code d'état une courte description en anglais du code (par exemple, 423 Locked). Les applications internationalisées ignoreront ce message, et afficheront un message approprié dans la langue et le jeu de caractères de l'utilisateur .

La présente spécification n'introduit aucune nouvelle chaîne qui soit affichée à l'utilisateur au titre du fonctionnement normal, sans erreur, du protocole.

Pour les raisons de ces décisions et des avis pour la mise en œuvre d'applications, voir la [RFC2518].

13. Considérations relatives à l'IANA

Le présent document utilise les espaces de noms définis dans la [RFC2518] pour les propriétés et les éléments XML. Toutes les autres considérations relatives à l'IANA mentionnées dans la [RFC2518] s'appliquent aussi au présent document.

14. Propriété intellectuelle

L'IETF ne prend pas position sur la validité et la portée de tout droit de propriété intellectuelle ou autres droits qui pourraient être revendiqués au titre de la mise en œuvre ou l'utilisation de la technologie décrite dans le présent document ou sur la mesure dans laquelle toute licence sur de tels droits pourrait être ou n'être pas disponible ; pas plus qu'elle ne prétend avoir accompli aucun effort pour identifier de tels droits. Les informations sur les procédures de l'ISOC au sujet des droits dans les documents de l'ISOC figurent dans les BCP 78 et BCP 79.

Des copies des dépôts d'IPR faites au secrétariat de l'IETF et toutes assurances de disponibilité de licences, ou le résultat de tentatives faites pour obtenir une licence ou permission générale d'utilisation de tels droits de propriété par ceux qui mettent en œuvre ou utilisent la présente spécification peuvent être obtenues sur répertoire en ligne des IPR de l'IETF à <http://www.ietf.org/ipr>.

L'IETF invite toute partie intéressée à porter son attention sur tous copyrights, licences ou applications de licence, ou autres droits de propriété qui pourraient couvrir les technologies qui peuvent être nécessaires pour mettre en œuvre la présente norme. Prière d'adresser les informations à l'IETF à ietf-ipr@ietf.org.

15. Contributeurs

Le présent document a bénéficié de contributions significatives de Geoff Clemm, Jason Crawford, Jim Davis, Chuck Fay et Judith Slein.

16. Remerciements

Le présent document a bénéficié de discussions intéressantes avec Jim Amsden, Steve Carter, Tyson Chihaya, Ken Coar, Ellis Cohen, Bruce Cragun, Spencer Dawkins, Mark Day, Rajiv Dulepet, David Durand, Lisa Dusseault, Roy Fielding, Yaron Goland, Fred Hitt, Alex Hopmann, Marcus Jager, Chris Kaler, Manoj Kasichainula, Rohit Khare, Daniel LaLiberte, Steve Martin, Larry Masinter, Jeff McAffer, Surendra Koduru Reddy, Max Rible, Sam Ruby, Bradley Sergeant, Nick Shelness, John Stracke, John Tighe, John Turner, Kevin Wiggen, et d'autres.

17. Références normatives

- [RFC2119] S. Bradner, "[Mots clés à utiliser](#) dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997.
- [RFC2277] H. Alvestrand, "Politique de l'IETF en matière de [jeux de caractères et de langages](#)", BCP 18, janvier 1998.
- [RFC2396] T. Berners-Lee, R. Fielding et L. Masinter, "Identifiants de ressource uniformes (URI) : Syntaxe générique", août 1998. (*Obsolète, voir [RFC3986](#)*)
- [RFC2518] Y. Goland, E. Whitehead, A. Faizi, S. Carter et D. Jensen, "Extensions [HTTP pour la création répartie](#) -- WEBDAV", février 1999. (*Obsolète, voir la RFC [4918](#)*)
- [RFC2616] R. Fielding et autres, "[Protocole de transfert hypertexte](#) -- HTTP/1.1", juin 1999. (*D.S., MàJ par [2817](#), [6585](#)*)
- [RFC3253] G. Clemm et autres, "[Extensions de versions à WebDAV](#) (Protocole de collecte ordonnée des auteurs et des versions distribuée sur la Toile)", mars 2002. (*P.S.*)
- [XML] Bray, T., Paoli, J., Sperberg-McQueen, C. and E. Maler, "Extensible Markup Language (XML) 1.0 (2nd ed)", W3C REC-xml, octobre 2000, < <http://www.w3.org/TR/2000/REC-xml-20001006> >.

Appendice A. Extensions à la définition de type de document WebDAV

```
<!ELEMENT orderpatch (ordering-type?, order-member*) >
<!ELEMENT order-member (segment, position) >
<!ELEMENT ordering-type (href) >
<!ELEMENT position (first | last | before | after)>
<!ELEMENT first EMPTY >
<!ELEMENT last EMPTY >
<!ELEMENT before segment >
<!ELEMENT after segment >
<!ELEMENT segment (#PCDATA)>
```

Index

Classe de conformité d'en-tête DAV 'ordered-collections'	Section 10
Collection non ordonnée	Section 3
Collection ordonnée	Section 3
En-têtes	
Ordering-Type	paragraphe 4.1.1
Position	Section 6

Méthodes	
ORDERPATCH	Section 7
Noms de conditions	
DAV:collection-must-be-ordered (pre)	paragraphe 6.1
DAV:initialize-collection-version-ordering-type (post)	paragraphe 9.2
DAV:initialize-ordering-type (post)	paragraphe 9.3
DAV:initialize-version-controlled-bindings-ordered (post)	paragraphe 9.2
DAV:initialize-version-history-bindings-ordered (post)	paragraphe 9.2
DAV:ordered-collections-supported (pre)	paragraphe 5.1
DAV:ordering-modified (post)	Section 7
DAV:ordering-type-set (post)	paragraphe 5.1, Section 7
DAV:position-set (post)	paragraphe 6.1
DAV:segment-must-identify-member (pre)	paragraphe 6.1
DAV:update-version-controlled-collection-members-ordered	paragraphe 9.2
DAV:update-version-ordering-type (post)	paragraphe 9.2
Ordre entretenu par le client	Section 3
Ordre entretenu par le serveur	Section 3
Postcondition DAV:initialize-collection-version-ordering-type	paragraphe 9.2
Postcondition DAV:initialize-ordering-type	paragraphe 9.3
Postcondition DAV:initialize-version-controlled-bindings-ordered	paragraphe 9.2
Postcondition DAV:initialize-version-history-bindings-ordered	paragraphe 9.2
Postcondition DAV:ordering-modified	Section 7
Postcondition DAV:ordering-type-set	paragraphe 5.1, Section 7
Postcondition DAV:position-set	paragraphe 6.1
Postcondition DAV:update-version-controlled-collection-members-ordered	paragraphe 9.2
Postcondition DAV:update-version-ordering-type	paragraphe 9.2
Précondition DAV:collection-must-be-ordered	paragraphe 6.1
Précondition DAV:ordered-collections-supported	paragraphe 5.1
Précondition DAV:segment-must-identify-member	paragraphe 6.1
Propriétés	
DAV:ordering-type	paragraphe 4.1.1
Sémantique de rangement	Section 3
Type d'ordre DAV:custom	paragraphe 4.1.1
Type d'ordre DAV:unordered	paragraphe 4.1.1

Adresse des auteurs

Jim Whitehead
 UC Santa Cruz, Dept. of Computer Science
 1156 High Street
 Santa Cruz, CA 95064
 US
 mél : ejw@cse.ucsc.edu

Julian F. Reschke, Ed.
 greenbytes GmbH
 Salzmanstrasse 152
 Muenster, NW 48159
 Germany
 téléphone : +49 251 2807760
 mél : julian.reschke@greenbytes.de

Déclaration complète de droits de reproduction

Copyright (C) The Internet Society (2003). Tous droits réservés.

Ce document et les traductions de celui-ci peuvent être copiés et diffusés, et les travaux dérivés qui commentent ou expliquent autrement ou aident à sa mise en œuvre peuvent être préparés, copiés, publiés et distribués, partiellement ou en totalité, sans restriction d'aucune sorte, à condition que l'avis de droits de reproduction ci-dessus et ce paragraphe soit inclus sur toutes ces copies et œuvres dérivées. Toutefois, ce document lui-même ne peut être modifié en aucune façon, par exemple en supprimant

le droit d'auteur ou les références à l'Internet Society ou d'autres organisations Internet, sauf si c'est nécessaire à l'élaboration des normes Internet, auquel cas les procédures pour les droits de reproduction définis dans les processus de normes pour Internet doivent être suivies, ou si nécessaire pour le traduire dans des langues autres que l'anglais.

Les permissions limitées accordées ci-dessus sont perpétuelles et ne seront pas révoquées par la Société Internet ou ses successeurs ou ayants droit.

Ce document et les renseignements qu'il contient sont fournis "TELS QUELS" et L'INTERNET SOCIETY et L'INTERNET ENGINEERING TASK FORCE déclinent toute garantie, expresse ou implicite, y compris mais sans s'y limiter, toute garantie que l'utilisation de l'information ici présente n'enfreindra aucun droit ou aucune garantie implicite de commercialisation ou d'adaptation à un objet particulier.

Remerciement

Le financement de la fonction d'édition des RFC est actuellement fourni par la Internet Society.