

4^e Dimension 6.7

Mise à jour
Windows[®]/Mac[™] OS



4e Dimension 6.7

Mise à jour

Copyright© 1985 - 2000 4D SA / 4D, Inc.
Tous droits réservés.

Les informations contenues dans ce manuel peuvent faire l'objet de modifications sans préavis et ne sauraient en aucune manière engager 4D SA. La fourniture du logiciel décrit dans ce manuel est régie par un octroi de licence dont les termes sont précisés par ailleurs dans la licence électronique figurant sur le support du Logiciel et de la Documentation y afférente. Le logiciel et sa Documentation ne peuvent être utilisés, copiés ou reproduits sur quelque support que ce soit et de quelque manière que ce soit, que conformément aux termes de cette licence.

Aucune partie de ce manuel ne peut être reproduite ou recopiée de quelque manière que ce soit, électronique ou mécanique, y compris par photocopie, enregistrement, archivage ou tout autre procédé de stockage, de traitement et de récupération d'informations, pour d'autres buts que l'usage personnel de l'acheteur, et ce exclusivement aux conditions contractuelles, sans la permission explicite de 4D SA.

4D, 4D Draw, 4D Write, 4D Insider, 4ème Dimension®, 4D Server, 4D Compiler, 4D Backup ainsi que les logos 4e Dimension et 4D sont des marques enregistrées de 4D SA.

Windows, Windows NT, Win32s et Microsoft sont des marques enregistrées de Microsoft Corporation.

Apple, Macintosh, Power Macintosh, LaserWriter, ImageWriter, QuickTime sont des marques enregistrées ou des noms commerciaux de Apple Computer, Inc.

Mac2Win Software Copyright © 1990-2000 est un produit de Altura Software, Inc.

ACROBAT © Copyright 1987-2000, Secret Commercial Adobe Systems Inc. Tous droits réservés. ACROBAT est une marque enregistrée d'Adobe Systems Inc.

Tous les autres noms de produits ou appellations sont des marques déposées ou des noms commerciaux appartenant à leurs propriétaires respectifs

Sommaire

Chapitre 1	Introduction	7
	A propos de ce manuel	7
	Compatibilité avec les versions précédentes	8
	ACI devient 4D	9
 Chapitre 2	 Mode Structure	 11
	Installation et gestion des fichiers 4 ^e Dimension	12
	Nouvelle architecture des fichiers 4D	12
	Transport direct des bases 4D	12
	Option d'installation de 4D Web Assistant	14
	Gestion des formulaires	15
	Héritage de formulaire	15
	Raccourci d'affichage de la page 0	17
	Edition des feuilles de style depuis la Liste des propriétés	17
	Interface de plate-forme	18
	Objets de formulaires	20
	Explorateur d'exécution	21
	Propriétés de la base	22
	Appel au système	22
	Connexions sécurisées (SSL)	23
	Nombre maximum de process Web	24
	Adresse IP d'écoute	25
	Racine HTML par défaut : DossierWeb	26
	Utilisation des composants 4D	27
	Présentation	27
	Visualisation et utilisation des composants dans 4D	27
 Chapitre 3	 Langage	 31
	Nouvelles commandes	32
	BLOB	32
	CRYPTER BLOB	32
	DECRYPTER BLOB	38
	Définition structure	39

LIRE PROPRIETES SAISIE CHAMP	39
LIRE PROPRIETES LIEN	40
LIRE PROPRIETES TABLE	42
Documents système	43
CREER ALIAS	43
RESOUDRE ALIAS	44
SUPPRIMER DOSSIER	45
LIRE ICONE DOCUMENT	45
Environnement 4D	46
LIRE INFORMATIONS SERIALISATION	46
Images	48
ECRIRE FICHER IMAGE	48
LIRE FICHER IMAGE	49
LISTE TYPES IMAGES	50
IMAGE VERS BLOB	52
BLOB VERS IMAGE	53
CREER IMAGETTE	54
Langage	56
Nom methode courante	56
Protocole sécurisé	57
GENERER CLES CRYPTAGE	57
GENERER DEMANDE CERTIFICAT	59
Ressources	62
Lire ID ressource composant	62
Serveur Web	63
Connexion Web securisee	63
ENVOYER TEXTE HTML	64
LIRE ENTETE HTTP	65
LIRE VARIABLES FORMULAIRE WEB	68
Commandes modifiées	70
CHANGER PROPRIETES LISTE	
(Thème Listes hiérarchiques)	70
COMPRESSER IMAGE (Thème Images)	70
CREER ENSEMBLE SUR TABLEAU (Thème Ensembles)	71
CREER SELECTION SUR TABLEAU (Thème Sélections temporaires)	71
DEPLACER OBJET (Thème Propriétés des objets)	72
Dossier ACI (Thème Environnement 4D)	72
Dossier systeme (Thème Environnement système)	72
FIXER ENTETE HTTP (Thème Serveur Web)	73
FIXER INTERFACE (Thème Interface utilisateur)	74
FIXER PARAMETRE BASE (Thème Définition structure)	75
LIRE IMAGE DANS BIBLIOTHEQUE (Thème Images)	80
Lire interface (Thème Interface utilisateur)	81
Lire parametre base (Thème Définition structure)	81

LIRE PROPRIETES CHAMP (Thème Définition structure)	82
LIRE PROPRIETES LISTE (Thème Listes hiérarchiques)	83
OUVRIR URL WEB (Thème Serveur Web)	83
PICT VERS GIF (Thème Images)	83
SIECLE PAR DEFALT (Thème Dates et heures)	84
SUPPRIMER IMAGE DANS BIBLIOTHEQUE (Thème Images)	84
TABLEAU ENTIER LONG SUR SELECTION (Thème Tableaux)	85
Types de fenêtres (Thème Fenêtres)	85

Chapitre 4	Serveur Web	87
	Compatibilité avec les technologies Internet	88
	WML	88
	HTML 4.0 et CSS1 - Feuilles de style en cascade, niveau 1	88
	XML	90
	Compatibilité avec les CGI	91
	Interface avec d'autres serveurs HTTP	96
	4DISAPI.DLL et NPH-CGI4D.EXE	96
	4D WebSTAR	100
	Gestion des pages HTML	101
	URL spécial /4DWEBTEST	101
	Suppression de la limite des 32000 caractères de texte	101
	Nouvelles balises HTML	102
	Optimisation du cache Web	110
	Connexions sécurisées via SSL	111
	Qu'est-ce que le protocole SSL ?	111
	Installation et activation de SSL dans 4D	112
	Obtenir un certificat SSL	114
	Connexions des browsers en SSL	116
Chapitre 5	Optimisations	119
	Editeur de commentaires	119
	Index, recherches et tris	119
	Recherches par contenu	119
	Tris	119
	Nombre de clés d'index	120
Index		121

1

Introduction

Bienvenue dans 4D version 6.7 ! Cette nouvelle version de 4^e Dimension et 4D Server apporte de nombreuses fonctionnalités supplémentaires, permettant de faciliter et d'accélérer le développement de vos bases de données 4D ainsi que leur publication sur le Web.

Ces nouveautés concernent principalement le mode Structure, le langage de programmation intégré et le serveur Web de 4D.

4D 6.7 propose un nouvel assistant permettant d'accélérer la publication Web de vos bases 4D, 4D Web Assistant.

En outre, les outils et plug-ins de l'environnement 4D 6.7 tels que 4D Insider, 4D Tools, 4D Internet Commands ou encore 4D Write proposent de nombreuses nouveautés.

Ces nouveautés font l'objet de documentations séparées.

A propos de ce manuel

Ce manuel détaille les nouveautés et modifications introduites dans la version 6.7 de 4^e Dimension et de 4D Server. Il se compose des chapitres suivants :

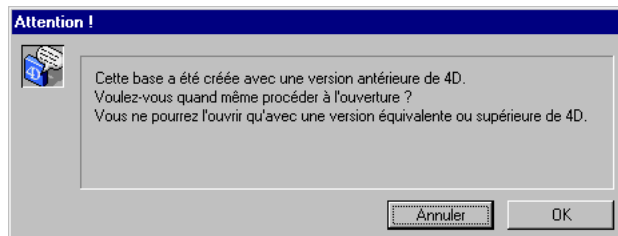
- **Mode Structure** Ce chapitre décrit les nouveautés et modifications concernant le mode Structure de 4D : nouvelle architecture et gestion des fichiers 4^e Dimension, modifications dans l'éditeur de formulaires, nouveautés concernant l'Explorateur d'exécution et les propriétés de la base, et utilisation des nouveaux "composants 4D" (générés à l'aide de 4D Insider).
- **Langage** Ce chapitre décrit les nouveautés et modifications concernant le langage de 4D : nouvelles commandes et commandes modifiées.

- **Serveur Web** Ce chapitre est consacré aux modifications apportées au serveur Web intégré de 4D : ces nouveautés concernent la compatibilité avec les technologies Internet et l'interfaçage avec d'autres serveurs HTTP, la gestion des pages HTML et le support des connexions sécurisées via SSL.
- **Optimisations** Ce chapitre décrit diverses optimisations apportées au moteur de base de données de 4D, notamment en ce qui concerne les recherches et tris indexés.

Compatibilité avec les versions précédentes

Les bases 4D créées avec des versions de 4D antérieures à la 6.7 (6.0.x et 6.5.x) sont compatibles avec 4D 6.7. Toutefois, le fichier de structure et le fichier de données seront convertis à l'ouverture de la base avec 4D 6.7. Une fois les fichiers convertis, vous ne pourrez plus les ouvrir avec une version précédente de 4D.

Lorsque vous ouvrez avec 4D 6.7 une base créée avec une version précédente de 4D, deux boîtes de dialogue successives vous avertissent que le fichier de structure et le fichier de données vont être convertis :



Dans les bases converties, les nouvelles options proposées par 4D 6.7 sont fixées de manière à ce que le fonctionnement initial de la base soit conservé. Ces options sont décrites au cours du texte.

ACI devient 4D

La société ACI a changé de raison sociale au mois d'Avril 2000, elle s'intitule désormais 4D.

En conséquence, les libellés (fichiers, dossiers, commandes...) comportant le nom de l'entreprise ont également été renommés en "4D". En particulier, le dossier "ACI", situé dans le dossier Système (sous Windows) et dans le dossier des Préférences (sous MacOS), s'intitule désormais "4D". La gestion par 4D de ce dossier est inchangée.

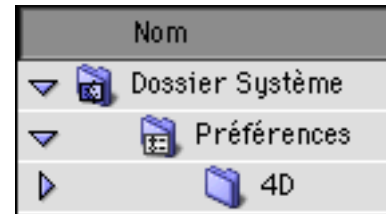
Ce dossier est automatiquement créé lorsque vous effectuez des installations à partir du CD-Rom 4D Product Line. En revanche, si vous mettez à jour votre environnement 4D à partir de fichiers téléchargés sur Internet, ou si vous utilisez des plug-ins fournis par sociétés tierces, vous devrez peut-être manuellement créer, mettre à jour ou configurer ce dossier 4D. Il doit être placé à côté du précédent dossier ACI :

- Sous Windows, à l'emplacement : C:\Windows\4D
- Sous MacOS, à l'emplacement : Disque:Système:Préférences:4D

Windows



MacOS



2

Mode Structure

Diverses modifications et nouveautés ont été apportées au mode Structure dans 4D version 6.7.

Ces nouveautés s'articulent autour des thèmes suivants :

- Installation et gestion des fichiers 4^e Dimension
- Gestion des formulaires
- Explorateur d'exécution
- Propriétés de la base
- Utilisation des composants 4D

Installation et gestion des fichiers 4^e Dimension

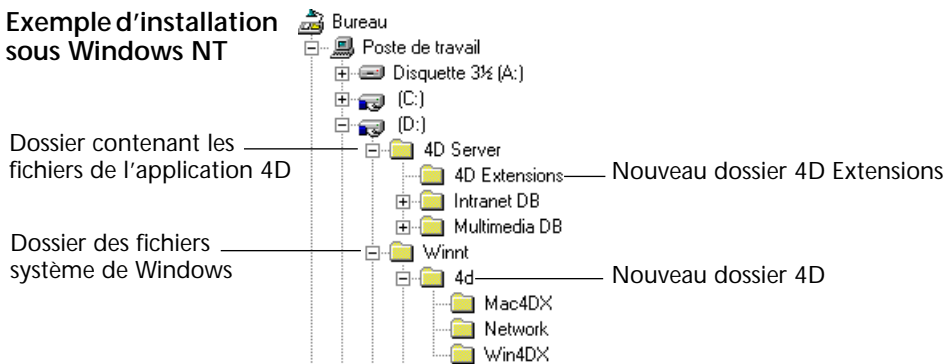
Nouvelle architecture des fichiers 4D

A compter de la version 6.7, les applications de la gamme 4^e Dimension s'appuient sur une nouvelle architecture de fichiers. Cette nouvelle architecture se traduit principalement par l'utilisation d'un dossier "4D Extensions" placé au même niveau que le fichier principal du programme. Veillez à ne pas déplacer ou renommer ce dossier. Il contient des fichiers de type "extensions de ressources", chargés au démarrage de l'application. Ces fichiers sont identifiés :

- sous MacOS, par leur type " 4DRS"
- Sous Windows, par leur extension ".4XR".

Note Suite au changement de raison sociale de la société ACI, les éléments auparavant nommés "ACI" ont été renommés "4D". En particulier, le dossier "ACI" (situé dans le dossier Système sous Windows et dans le dossier des Préférences sous MacOS) s'intitule désormais "4D". Pour plus d'informations sur ce point, reportez-vous au [paragraphe "Compatibilité avec les versions précédentes"](#), page 8.

Exemple d'installation sous Windows NT



Note Le mécanisme de téléchargement des fichiers dans le cadre d'une configuration Client/Serveur a été légèrement modifié. Pour plus d'informations, reportez-vous au *Guide d'installation*.

Transport direct des bases 4D

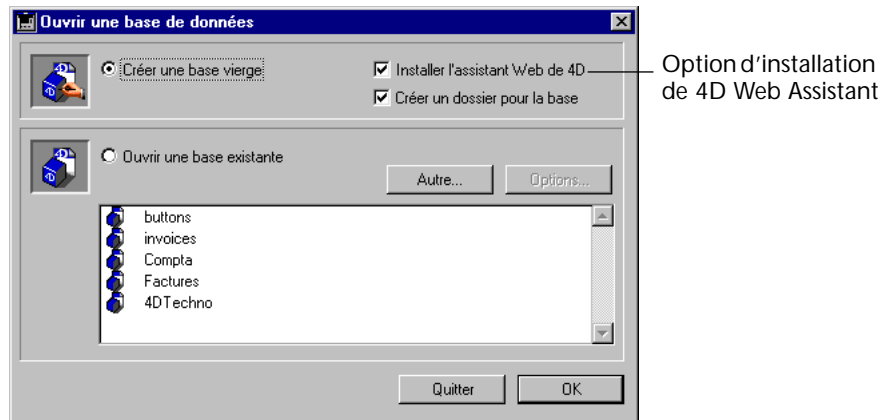
La version 6.7 de 4^e Dimension permet désormais de copier directement les fichiers des bases de la plate-forme Windows vers la plate-forme MacOS et inversement — il n'est plus nécessaire de transporter la base à l'aide de 4D Transporter. Attention, seuls les fichiers de structure (compilés ou non) et de données bénéficient de cette possibilité.

MacOS vers Windows	Une base 4D créée sur la plate-forme MacOS et copiée sur un volume Windows (formaté en NTFS ¹) peut être ouverte directement avec la version Windows de 4 ^e Dimension. L'application 4D effectue elle-même la séparation entre les parties "ressources" et "données" des fichiers.
<i>Note</i>	Les fichiers copiés sous Windows n'ont pas d'extension. Vous devrez donc saisir "*" dans la boîte de dialogue d'ouverture de fichiers afin de pouvoir les visualiser. De plus, ils ne seront pas double-cliquables.
Windows vers MacOS	<p>A l'inverse, une base 4D créée sur la plate-forme Windows et copiée sur un Macintosh peut être ouverte directement avec la version MacOS de 4^e Dimension, sans transport préalable. L'ouverture s'effectue de la manière suivante : en premier lieu, 4D ouvre les fichiers .4DB, .4DD ou .4DC et recherche les fichiers .RSR ou .4DR correspondants. S'il en trouve, le programme affiche une boîte de dialogue demandant à l'utilisateur de confirmer l'adéquation entre les deux fichiers. Une fois la boîte de dialogue validée, 4D réunit les fichiers de ressources et de données.</p> <p>Sous MacOS, 4D vérifie qu'un même fichier .RSR n'est pas utilisé par deux parties "données" (.4DB et .4DC). Si c'est le cas, le programme duplique le fichier .RSR avant de procéder à la réunion des fichiers.</p>

1. NTFS est un "système de fichiers", disponible sous Windows NT4 et Windows 2000 uniquement. Pour configurer un volume en NTFS, il est nécessaire de le reformater.

Option d'installation de 4D Web Assistant

La boîte de dialogue de création et d'ouverture des bases 4D comporte une nouvelle option : Installer l'assistant Web de 4D.



Note 4D Web Assistant facilite la publication Web de vos bases 4D. Cet outil fait l'objet d'une documentation séparée : pour plus d'informations, reportez-vous au manuel *4D Web Assistant*.

L'option "Installer l'assistant Web de 4D" n'est active qu'en cas de création d'une nouvelle base. Lorsqu'elle est cochée, 4D intègre le nouvel assistant Web dans la base créée.

A noter que, si vous ne cochez pas cette option, vous pourrez toutefois installer 4D Web Assistant à tout moment dans votre base de données. En effet, cet outil est composé exclusivement d'objets 4D et se présente sous la forme d'un *composant 4D* (pour plus d'informations sur ce point, reportez-vous au [paragraphe "Utilisation des composants 4D", page 27](#)).

L'installation d'un composant au sein d'une base 4D existante s'effectue à l'aide de 4D Insider.

Gestion des formulaires

Héritage de formulaire

Principe

La version 6.7 de 4^e Dimension introduit la notion d’“héritage de formulaire”. Le principe de cette nouveauté consiste à pouvoir utiliser dans un formulaire B tous les objets d’un formulaire A : le formulaire B “hérite” des objets du formulaire A.

Supposons par exemple que tous les formulaires de saisie d’une base doivent contenir les boutons OK, Annuler, Suivant et Précédent ainsi qu’un logo. Il vous suffit de créer un formulaire ne contenant que ces éléments, puis de l’appeler en tant que formulaire hérité dans tous les formulaires de saisie de la base. Chaque formulaire de saisie ne contient, quant à lui, que les champs et objets spécifiques à son utilisation.

A la différence des “modèles” de formulaires définis à l’aide de l’assistant de création de formulaires, la référence au formulaire hérité est toujours active : si un élément du formulaire hérité est modifié (par exemple le style des boutons), tous les formulaires qui l’utilisent seront automatiquement modifiés.

Fonctionnement

Lors de l’utilisation de la base, les objets du formulaire hérité sont combinés dynamiquement à ceux du formulaire ouvert. Ce mécanisme est très proche de celui de la “page zéro” des formulaires, à la différence qu’il peut s’appliquer à l’ensemble des formulaires de la base.

A l’ouverture du formulaire en mode Utilisation ou Menus créés, les objets sont chargés et combinés dans l’ordre suivant :

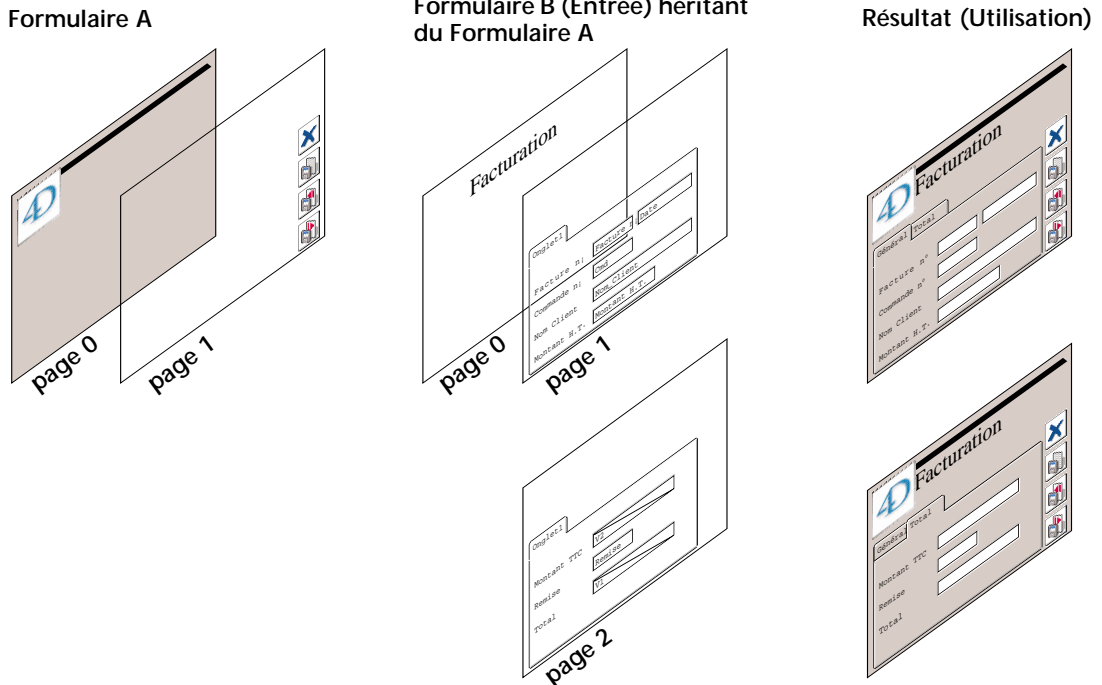
- 1 - Page zéro du formulaire hérité
- 2 - Page 1 du formulaire hérité
- 3 - Page zéro du formulaire ouvert
- 4 - Page courante du formulaire ouvert.

Cet ordre détermine l’ordre de saisie des objets dans le formulaire.

Note Seules les pages 0 et 1 du formulaire hérité peuvent apparaître dans les autres formulaires.

Les propriétés (nom de la fenêtre, options de redimensionnement, événements, etc.) ainsi que la méthode d'un formulaire hérité ne sont pas prises en compte lorsque celui-ci est utilisé comme formulaire hérité. En revanche, les méthodes des objets qu'il contient sont appelées.

Le schéma suivant illustre le mécanisme des formulaires hérités :



Définir un formulaire hérité

L'héritage d'un formulaire s'effectue à partir de l'éditeur de formulaires de 4D.

- Pour définir un formulaire hérité :
 - 1 Dans l'éditeur de formulaires, ouvrez le formulaire devant hériter d'un autre formulaire.
 - 2 Affichez la Liste des propriétés et cliquez en-dehors de tout objet du formulaire afin de visualiser les propriétés du formulaire.
Deux nouvelles propriétés sont disponibles : "Table du formulaire hérité" et "Nom du formulaire hérité".
 - 3 Sélectionnez la table puis le nom du formulaire duquel hériter.

Tout formulaire peut être désigné comme formulaire hérité. Toutefois, les éléments qu'il contient doivent être compatibles avec une utilisation dans différentes tables de la base.

Dès qu'un formulaire hérité est sélectionné, son contenu apparaît dans la fenêtre d'édition du formulaire courant. Il s'agit d'une prévisualisation, il n'est pas possible sélectionner ni de modifier un objet de ce formulaire. Pour cela, vous devez l'ouvrir dans sa propre fenêtre.

Vous pouvez masquer les objets d'un formulaire hérité en désélectionnant l'option **Formulaire hérité** dans le sous-menu **Afficher** du menu **Formulaire** ou du menu contextuel de l'éditeur.

Pour stopper l'héritage d'un formulaire, choisissez l'option **<Aucun>** dans la Liste des propriétés.

Note Il est possible de définir un formulaire hérité dans un formulaire qui servira à son tour de formulaire hérité pour un troisième formulaire. La combinaison des objets s'effectue alors de manière récursive. 4D détecte toutefois les récursions en boucle (par exemple si le formulaire [table1]form1 est défini comme formulaire hérité de [table1]form1, c'est-à-dire de lui-même) et interrompt le chaînage des formulaires.

Raccourci d'affichage de la page 0

Dans l'éditeur de formulaires, un nouveau raccourci permet d'afficher directement la page 0 du formulaire courant à partir de toute page du formulaire : pour cela, effectuez **Alt+clik** sous Windows ou **Option+clik** sous MacOS sur un objet appartenant à la page 0, ou plus exactement hors de tout objet appartenant à la page courante (le raccourci **Alt+clik** ou **Option+clik** sur un objet de la page courante crée ou ouvre la méthode de l'objet).

L'éditeur de formulaires affiche alors automatiquement la page zéro du formulaire.

Edition des feuilles de style depuis la Liste des propriétés

Il est désormais possible d'afficher la boîte de dialogue des propriétés des feuilles de style depuis la Liste des propriétés. Il suffit pour cela de cliquer sur le bouton situé à côté de la liste déroulante de sélection des feuilles de style.

Interface de plate-forme

Les différentes options d'interface ont été réorganisées et une nouvelle option est disponible : Thème Mac. En outre, le fonctionnement de l'option d'interface de plate-forme Automatique a été légèrement modifié.

Choix d'interface

- L'option d'interface MacOS est renommée en Mac OS 7 afin de mieux indiquer le type d'apparence représenté.
- Une nouvelle option d'interface est disponible : Thème Mac¹. Lorsque cette option est sélectionnée, les formulaires de 4D prennent automatiquement l'apparence du thème défini dans le tableau de bord "Apparence" de MacOS.

Ce type d'interface influe sur tous les types d'objets actifs des formulaires : boutons standard, zones de défilement, listes déroulantes, etc. En outre, il permet d'utiliser des onglets orientés à droite, à gauche ou en bas (cf. [paragraphe "Orientation des onglets \(MacOS uniquement\)"](#), page 20).

Lorsque l'option Thème Mac est sélectionnée pour un formulaire affiché sous Windows, l'apparence standard "Windows 9x" est appliquée.

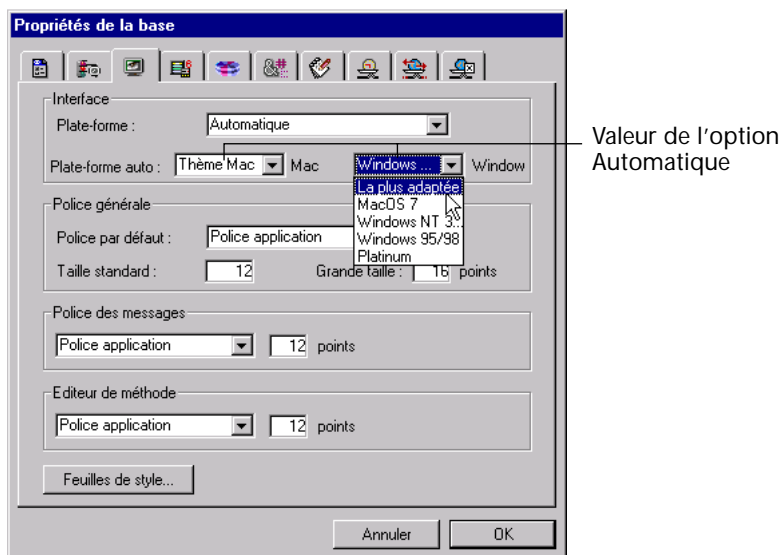
Le fonctionnement de l'option Thème Mac dans 4D n'est garanti qu'avec un thème fourni par Apple.

Cette nouvelle option peut également être définie via la commande [FIXER INTERFACE](#).

1. Cette fonctionnalité n'est disponible qu'à partir de la version 1.1 du "Gestionnaire d'apparence".

Option "Automatique"

La gestion de l'option Automatique a été modifiée dans 4D 6.7. Désormais, dans la page "Interface utilisateur" des Propriétés de la base, deux pop up menus permettent de définir précisément l'interface utilisée, pour chaque plate-forme, lorsque l'option Automatique est sélectionnée :



Chaque menu comporte l'option **La plus adaptée**. Lorsque cette option est sélectionnée, 4D choisit pour la plate-forme automatique l'interface la plus "moderne" en fonction des capacités de la machine sur laquelle le programme est exécuté :

- Sous Windows, l'interface **Windows 95/98** est utilisée, quelle que soit la version de Windows.
- Sous MacOS, l'interface **Thème Mac** est généralement utilisée, sauf si l'application est exécutée sur un système trop ancien, auquel cas l'interface **Platinum** est utilisée.
- **Conversion des bases 6.5.x**
Lorsque l'option "Automatique" était utilisée dans des bases 4D 6.5 converties en version 6.7, 4D fixe les nouvelles valeurs de la manière suivante :
 - Sous Windows, l'option **La plus adaptée** est sélectionnée.
 - Sous MacOS, l'option **La plus adaptée** est sélectionnée si la case "Utiliser Platinum en automatique" était cochée. Sinon, l'option **Mac OS 7** est sélectionnée.

Objets de formulaires

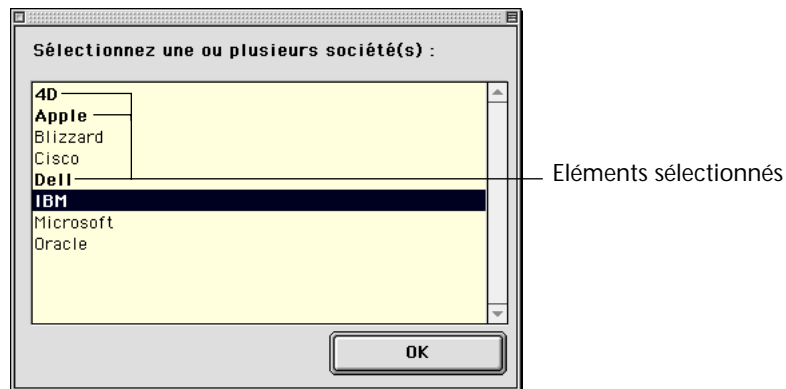
Variables non-saisissables et clics

Les événements formulaire Sur clic souris et Sur double clic souris peuvent désormais être utilisés avec les variables de formulaire ayant l'attribut "non saisissable".

Ce fonctionnement facilite en particulier la gestion des menus contextuels dans les formulaires 4D.

Zones de défilement transparentes

Il est désormais possible de rendre transparente une zone de défilement insérée dans un formulaire. Cette nouveauté permet par exemple de créer des interfaces personnalisées simulant des sélections multiples dans ce type de zone :



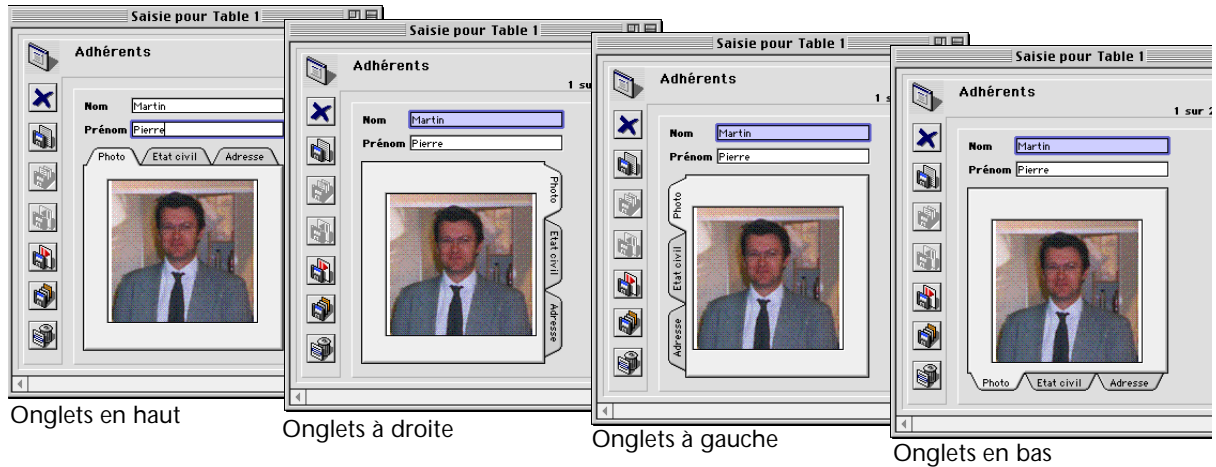
Dans l'exemple ci-dessus, deux zones de défilement sont superposées. La zone située au premier plan dispose de l'attribut "Transparent" et charge par défaut la liste des valeurs. La zone d'arrière-plan, vide par défaut, comporte un style de police particulier (gras). Suivant la ligne sur laquelle l'utilisateur clique, l'élément de l'une ou de l'autre zone est affiché, simulant ainsi une sélection/désélection.

Orientation des onglets (MacOS uniquement)

Vous pouvez désormais modifier l'emplacement des onglets dans vos formulaires. Cette fonctionnalité est disponible à deux conditions :

- Le formulaire est affiché sous MacOS (version 8 minimum).
- La propriété d'apparence "Plate-forme" appliquée à l'objet est Thème Mac (pour plus d'informations sur ce point, reportez-vous ci-dessus au [paragraphe "Choix d'interface", page 18](#))

Lorsque ces deux conditions sont réunies, vous pouvez choisir de placer les onglets en haut (standard), à droite, à gauche ou en bas :



Lorsqu'un formulaire comportant des onglets personnalisés est affiché sous Windows ou sous MacOS avec une apparence autre que "Thème Mac", les onglets prennent l'emplacement standard.

Explorateur d'exécution

La page Evaluation (rubrique "Informations") de l'Explorateur d'exécution affiche trois nouvelles informations :

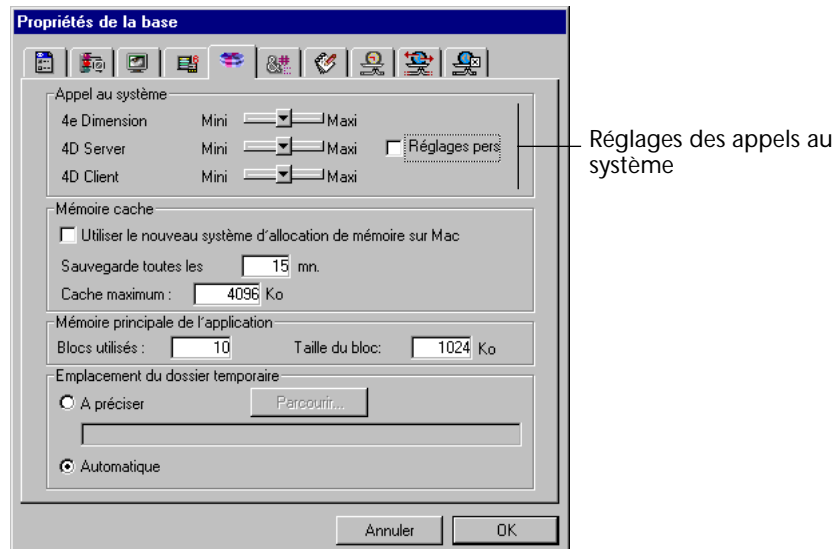
- **Occupation du cache Web** : indique le nombre de pages présentes dans le cache Web ainsi que son pourcentage d'utilisation. Cette information n'est disponible que si le serveur Web est actif et si la taille du cache différente de 0.
- **Temps d'activité du serveur Web** : indique la durée de fonctionnement (au format heures:minutes:secondes) du serveur Web. Cette information n'est disponible que si le serveur Web est actif.
- **Nombre de requêtes HTTP** : indique le nombre total de requêtes HTTP reçues depuis le démarrage du serveur Web, ainsi que le nombre instantané de requêtes par secondes (mesure prise entre deux mises à jour de l'Explorateur d'exécution). Cette information n'est disponible que si le serveur Web est actif.

Propriétés de la base

Appel au système

Le paramétrage du gestionnaire 4D d'appels au système en mode interprété (nombre de ticks entre deux appels...) a été simplifié dans 4D 6.7.

Désormais, le réglage se limite à une seule donnée, indiquant la priorité allouée à 4D. Les valeurs possibles sont “minimum”, “moyen” et “maximum”. Il s'effectue dans la page “Réglages système” de la boîte de dialogue des Propriétés de la base :



Chaque valeur correspond à un groupe de paramétrages incluant :

- le nombre de ticks entre deux appels au système,
- le nombre minimum de ticks pour chaque appel,
- le nombre maximum de ticks pour chaque appel.

Pour une même base, il est possible de régler distinctement la priorité pour 4^e Dimension (monoposte)/4D Tools, 4D Server et 4D Client. En outre, il est possible de connaître et de fixer chaque réglage par programmation, à l'aide des commandes [Lire parametre base](#) et [FIXER PARAMETRE BASE](#).

Par défaut, les valeurs initiales d'une base correspondent aux réglages “moyens”.

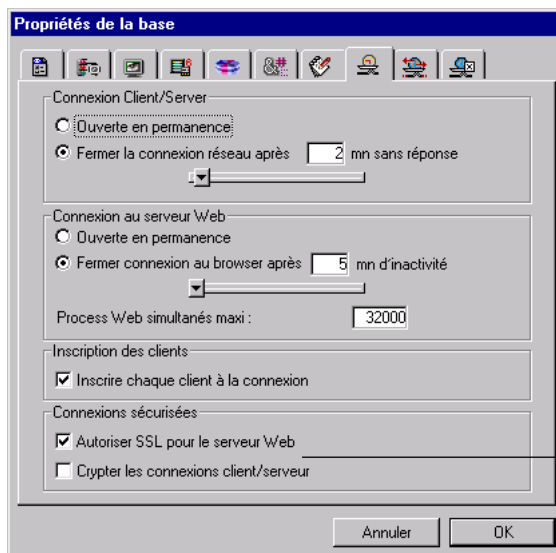
Compatibilité avec les bases précédentes

Si, pour les applications 4^e Dimension (monoposte), 4D Server et 4D Client, une base développée avec une version précédente avait des réglages de priorité standard, les trois thermomètres sont actifs avec 4D 6.7.

Si, pour une de ces applications, les réglages avaient été modifiés, l'option Réglages personnalisés est cochée. Dans ce cas, les thermomètres sont inactivés. Ainsi, les précédentes valeurs personnalisées sont conservées.

Connexions sécurisées (SSL)

La page “Connexions” des Propriétés de la base comporte deux nouvelles options permettant de définir l'utilisation du protocole de sécurisation SSL (*Secured Socket Layout*) dans le cadre des connexions réseau. Pour plus d'informations sur le protocole SSL, reportez-vous au [paragraphe “Connexions sécurisées via SSL”, page 111](#).



Paramétrage du protocole sécurisé SSL

- **Autoriser SSL pour le serveur Web** : permet d'activer ou d'inactiver l'utilisation du protocole SSL pour les connexions du serveur Web. Par défaut, cette option est cochée. Le port TCP utilisé pour les connexions SSL est le port 443.

Vous pouvez désélectionner cette option si vous ne souhaitez pas exploiter les fonctionnalités SSL avec votre serveur Web, ou si un autre serveur Web autorisant les connexions sécurisées fonctionne sur le même poste.

- **Crypter les connexions client/serveur** : permet d'activer ou d'inactiver le cryptage des connexions 4D Server. En effet, l'architecture client/serveur "classique" peut tirer parti des fonctionnalités de cryptage proposées par le protocole SSL. Ce fonctionnement permet de renforcer la sécurité des communications, mais ralentit les connexions. A noter que cette option ne nécessite aucun paramétrage supplémentaire (cf . [paragraphe "Paramétrer SSL en client/serveur"](#), page 113.)
Par défaut, l'option n'est pas cochée.

Nombre maximum de process Web

La page "Connexions" des Propriétés de la base permet de définir une nouvelle valeur : la limite strictement supérieure du nombre de Process Web simultanés maxi. Ce paramètre permet de prévenir la saturation du serveur Web 4D pouvant se produire lors d'un envoi massif de requêtes ou d'une demande excessive de création de contextes.

Ce paramètre définit le nombre maximum de process Web de tout type : contextuels, non contextuels ou appartenant à la "réserve" de process. Par défaut, ce nombre est de 32 000 (autrement dit, jusqu'à 31 999 process Web peuvent être créés simultanément). Vous pouvez passer toute valeur incluse entre 10 et 32 000.

Lorsque ce nombre maximum (moins un) de process Web concurrents est atteint, 4D ne crée plus de nouveau process et retourne le message "Serveur non disponible" (statut HTTP 503 - Service Unavailable) à toute nouvelle requête.

Comment déterminer la valeur à passer ?

En théorie, le nombre maximum de process Web est le résultat de la division Mémoire disponible / Taille de la pile d'un process Web.
Une autre solution consiste à visualiser les informations sur les process Web affichées dans l'Explorateur d'exécution : le nombre courant de process Web et le nombre maximum atteint depuis le démarrage du serveur Web sont indiqués.

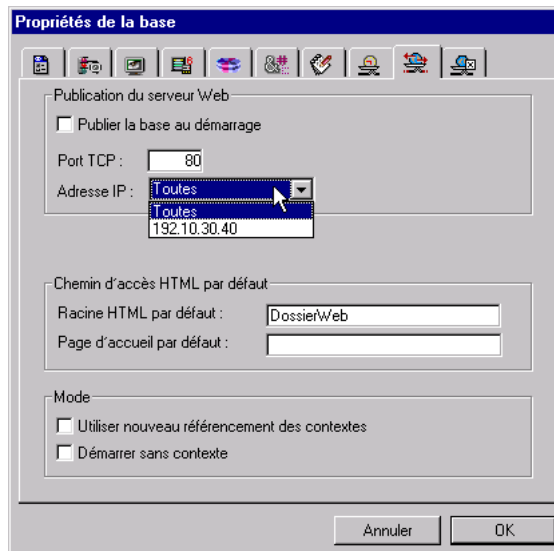
Réserve de process Web

La "réserve" de process Web permet d'augmenter la réactivité du serveur Web en mode sans contexte. Cette réserve est dimensionnée par un minimum (0 par défaut) et un maximum (10 par défaut) de process à recycler. Ces valeurs peuvent être modifiées à l'aide de la commande FIXER PARAMETRE BASE (sélecteurs 6 et 7). Lors du changement du nombre maximum de process Web, si celui-ci est inférieur à la limite supérieure de la "réserve", cette limite est alors abaissée au nombre maximum de process Web.

Le nombre maximum de process Web peut également être défini à l'aide de la commande `FIXER PARAMETRE BASE`. Pour plus d'informations, reportez-vous à la description de la [routine FIXER PARAMETRE BASE \(Thème Définition structure\)](#), page 75.

Adresse IP d'écoute

Le mode de définition de l'adresse IP d'écoute du serveur Web a été simplifié. La page "Serveur Web I" de la fenêtre des Propriétés de la base comporte désormais un pop up menu listant automatiquement toutes les adresses IP présentes sur la machine. Ce menu comporte également l'option Toutes (sélectionnée par défaut), indiquant que le serveur répond sur toutes les adresses IP.



Si vous souhaitez que le serveur ne réponde qu'aux requêtes adressées sur une adresse IP particulière, il suffit de sélectionner cette adresse dans le menu.

Racine HTML par défaut : DossierWeb

Désormais, lorsque vous créez une nouvelle base de données avec 4D 6.7, le programme définit un dossier racine HTML par défaut dans la boîte de dialogue des Propriétés de la base, nommé “DossierWeb”.

Ce fonctionnement est destiné à renforcer la sécurité du serveur Web 4D. Rappelons en effet que le dossier racine HTML indique, sur le disque dur du serveur Web, le niveau hiérarchique au-dessus duquel l'accès est protégé, et qu'en cas d'absence de ce paramètre, aucune restriction d'accès n'est appliquée (voir à ce sujet la section “Services Web, Sécurité des connexions” dans le manuel *Langage* de 4D).

Ce nouveau paramétrage par défaut active donc automatiquement le système de restrictions d'accès pour les nouvelles bases 4D.

Note Les bases créées avec une version antérieure de 4D et converties en version 6.7 conservent leurs paramètres précédents.

Le dossier “DossierWeb” n'est pas créé physiquement sur le disque. Pour que les mécanismes du serveur Web 4D exploitant le dossier racine HTML par défaut fonctionnent, vous devez créer un dossier “DossierWeb” et le placer au niveau du fichier de structure de la base — bien entendu, vous pouvez modifier son nom et son emplacement dans la boîte de dialogue des Propriétés de la base. Il vous suffit ensuite de copier les éléments requis (pages statiques, images...) dans ce dossier.

Utilisation des composants 4D

Présentation

La version 6.7 de 4^e Dimension et de 4D Insider permettent le développement et la diffusion de *composants*. Schématiquement, un composant regroupe un ensemble d'objets structurels 4D (tables, méthodes, formulaires, barres de menus...) représentant une ou plusieurs fonctionnalités supplémentaires. Par exemple, vous pouvez développer à l'aide de 4D un composant de courrier électronique. Un composant est autonome, il doit pouvoir être installé dans tout fichier de structure 4D.

Les composants 4D sont construits, générés et installés à l'aide de 4D Insider. A la différence des *librairies* et des *groupes* (auxquels ils s'apparentent), les composants intègrent la notion de *protection* des objets qui les composent. Chaque objet d'un composant se voit attribuer le type "public", "protégé" ou "privé", ce qui détermine s'il pourra être visible ou modifiable une fois le composant installé. Les composants ont pour but de permettre aux développeurs de commercialiser des solutions originales en toute sécurité.

Pour une description détaillée de la création et de l'installation des composants, veuillez vous reporter au Guide de référence de 4D Insider. Nous ne traitons ici que l'utilisation des composants au sein des bases 4D.

Visualisation et utilisation des composants dans 4D

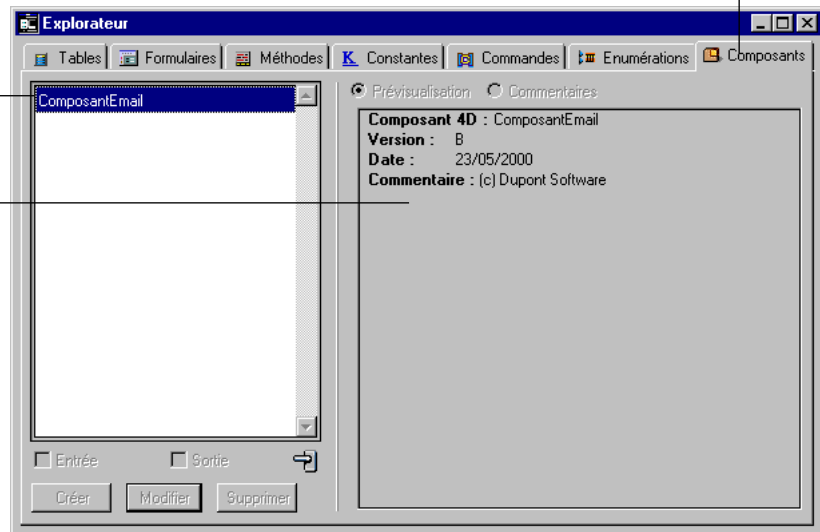
Lorsqu'un composant est installé dans une base 4D, tous les objets qu'il contient sont ajoutés dans la base et sont disponibles dans les divers éditeurs de 4^e Dimension — lorsque leur type est "Public" ou "Protégé" (cf. [paragraphe "Attributs des objets", page 29](#)).

Chaque composant 4D installé dans la base est listé dans la page Composants de l'Explorateur de 4D :

Onglet Composants

Liste des composants installés dans la base

Propriétés du composant sélectionné (fournies par le développeur du composant)



Les propriétés du composant apparaissent dans la zone de prévisualisation de l'Explorateur, lorsque celle-ci est déployée : Nom du composant, Version, Date de création et Commentaires. Ces informations sont fournies par le développeur du composant.

Types d'objets

La plupart des objets composant une base 4D peuvent être inclus dans un composant :

- Tables et champs
- Triggers
- Formulaires
- Méthodes formulaire, Méthodes objet et Méthodes projet
- Menus et barres de menus
- Enumérations
- Info-bulles
- Ressources (STR# et PICT)
- Images de la bibliothèque
- Formats et filtres
- Feuilles de style
- Commentaires

Les objets de langage suivants peuvent également être inclus dans un composant :

- Variables locales, process et interprocess
- Ensembles
- Sélections
- Sémaphores

Attributs des objets

Lors de la création d'un composant avec 4D Insider, chaque objet inclus dans le composant reçoit l'un des trois attributs suivants : **Public**, **Protégé** ou **Privé**. Ces attributs, à la base du système de protection des composants, indiquent si les objets concernés seront visibles et modifiables dans le mode Structure de 4^e Dimension et dans 4D Insider, une fois le composant généré et installé.

- **Public** : les objets “publics” seront visibles et modifiables par les utilisateurs, toutefois ils ne pourront être ni renommés ni supprimés. Ce type d'objet peut être utile pour fournir des objets personnalisables par les utilisateurs. Dans les éditeurs de 4D, les objets publics apparaissent comme tous les autres objets.
- **Protégé** : les objets protégés seront visibles mais ne pourront être ni modifiés ni supprimés par les utilisateurs. Une méthode protégée peut être appelée, mais son contenu ne peut être ni visualisé ni modifié (la zone de prévisualisation de l'Explorateur reste vide). Dans les éditeurs de 4D, l'icône des objets protégés est barrée d'un trait rouge :



MyMod_Prot_CallsTable1

- **Privé** : les objets privés ne seront ni visibles ni, par conséquent, modifiables pour les utilisateurs des composants, aussi bien dans 4^e Dimension que dans 4D Insider.

Le tableau suivant résume les possibilités offertes dans 4D et 4D Insider par les objets contenus dans les composants, en fonction de leur attribut :

	Nom visible	Contenu visible	Contenu modifiable	Renommable ou supprimable
<i>Public</i>	Oui	Oui	Oui	Non
<i>Protégé</i>	Oui	Non	Non	Non
<i>Privé</i>	Non	Non	Non	Non

Note De manière générale, les attributs des objets des composants seront respectés par toutes les applications et plug-ins de l'environnement 4D, tels que 4D Write ou 4D Compiler.

3

Langage

De nouvelles commandes ont été ajoutées dans 4^e Dimension et 4D Server version 6.7. Le comportement de certaines commandes existantes a été modifié.

Ces nouveautés et modifications liées au langage de 4D sont décrites dans ce chapitre.

Nouvelles commandes

BLOB

Les deux nouvelles commandes du thème BLOB permettent de crypter et de décrypter des données dans une base 4D. Ces commandes utilisent l’algorithme et les fonctions de cryptage du protocole SSL (proposé dans 4^e Dimension à compter de la version 6.7). Par conséquent, pour pouvoir utiliser ces commandes, vous devez veiller à ce que les composants nécessaires au fonctionnement du protocole SSL soient installés sur la machine — même si vous ne souhaitez pas utiliser SSL dans le cadre de connexions à un serveur Web 4D¹. Pour plus d’informations sur l’installation de ce protocole, reportez-vous au [paragraphe “Connexions sécurisées via SSL”, page 111](#).

CRYPTER BLOB

CRYPTER BLOB (aCrypter; cléPrivEmetteur{; cléPubRécepteur})

Paramètres	Type	Description
aCrypter	BLOB	→ Données à crypter
		← Données cryptées
cléPrivEmetteur	BLOB	→ Clé privée de l’émetteur
cléPubRécepteur	BLOB	→ Clé publique du récepteur

Cette commande permet de crypter le contenu du BLOB aCrypter à l’aide de la clé privée de l’émetteur cléPrivEmetteur ainsi que, optionnellement, de la clé publique du récepteur cléPubRécepteur.

Note Pour obtenir une paire de clés de cryptage (clé publique et clé privée), utilisez la [routine GENERER CLES CRYPTAGE, page 57](#), placée dans le thème “Protocole sécurisé”.

- L’utilisation d’une seule clé pour le cryptage (clé privée de l’émetteur) garantit l’impossibilité pour toute personne ne disposant pas de la clé publique de lire les données. Elle garantit également que c’est bien l’émetteur qui a crypté les données.
- L’utilisation d’une paire de clés pour le cryptage (clé privée de l’émetteur + clé publique du récepteur) garantit en outre qu’un seul récepteur pourra lire les données.

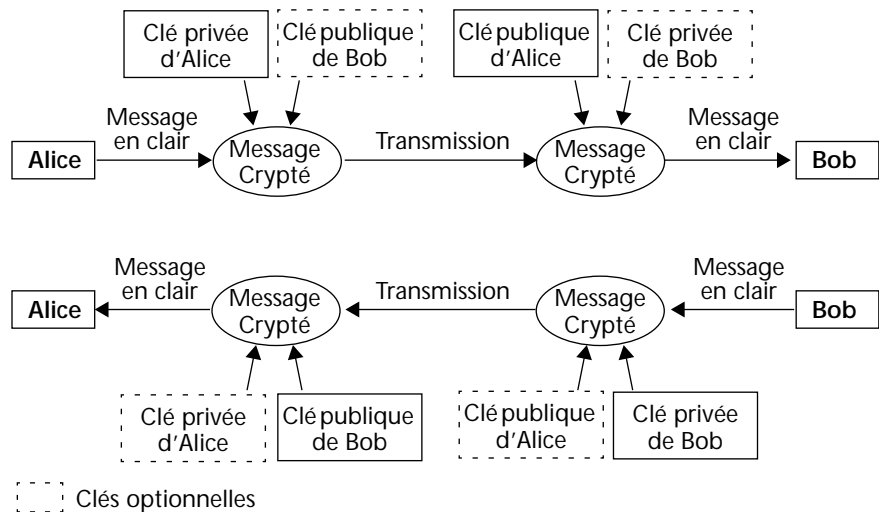
1. Une nouvelle option des Propriétés de la base permet de désactiver le mode SSL pour le serveur Web 4D.

Le format interne des BLOBs contenant des clés est le PEM (*Privacy Enhanced Mail*). Ce format, multi-plate-forme, permet l'échange ou la manipulation des clés par simple copier-coller dans un Email ou un fichier texte.

Après l'exécution de la commande, le BLOB aCrypter contient les données cryptées. Ces données ne pourront être décryptées qu'avec la commande DECRYPTER BLOB, à laquelle la clé publique de l'émetteur sera passée en paramètre.

En outre, si la clé publique (optionnelle) du récepteur avait été utilisée pour le cryptage, la clé privée du récepteur sera également nécessaire pour le décryptage.

Principe du cryptage à clés publiques/privées pour l'échange de messages entre deux individus, "Alice" et "Bob"



Note L'algorithme de cryptage comporte une fonction de vérification d'intégrité (*checksum*), afin d'empêcher toute modification malveillante ou accidentelle du contenu du BLOB. Par conséquent, un BLOB crypté ne doit pas être modifié, sous peine de ne pas pouvoir être décrypté.

Optimisation des commandes de cryptage

Le cryptage des données ralentit l'exécution de l'application, en particulier si une paire de clés est utilisée. Deux types d'optimisations sont toutefois possibles :

- Suivant la quantité de mémoire disponible, la commande s'exécute en mode "synchrone" ou "asynchrone".
Le mode asynchrone est plus rapide, car il ne bloque pas les autres process. Ce mode est automatiquement utilisé si la mémoire disponible est au moins égale à 2 fois la taille de la source à crypter.
Dans le cas contraire, pour des raisons de sécurité, le mode synchrone est utilisé. Ce mode est plus lent car les autres process sont bloqués.
- Dans le cas de BLOBs volumineux, l'astuce consiste à crypter uniquement une partie déterminée et sensible du BLOB, afin de réduire la taille des données à traiter et donc le temps d'exécution.

▼ Exemple 1 : Utilisation d'une seule clé

Une société veut garantir la confidentialité d'informations stockées dans une base 4D. Elle doit régulièrement envoyer ces données à ses filiales, par exemple sous la forme de fichiers via Internet.

1) La société commence par générer une paire de clés à l'aide de la commande GENERER CLES CRYPTAGE.

```
`Méthode GENERE_CLES_TXT  
C_BLOB($BcléPublique; $BcléPrivée)  
GENERER CLES CRYPTAGE($BcléPrivée;$BcléPublique)  
BLOB VERS DOCUMENT("cléPublique.txt"; $BcléPublique)  
BLOB VERS DOCUMENT("cléPrivée.txt"; $BcléPrivée)
```

2) La société conserve la clé privée, et remet à chaque filiale une copie du document contenant la clé publique. Il faut, bien entendu, que cette transmission s'effectue d'une façon sûre, par exemple par la copie sur une disquette donnée physiquement aux filiales.

3) Par la suite, la société copie les informations confidentielles (stockées par exemple dans un champ texte) dans des BLOBs et les crypte avec sa clé privée :

```
`Méthode CRYPTER_INFOS  
C_BLOB($vbCrypté;$vbcléPrivée)  
C_TEXTE($vtCrypter)  
  
$vtCrypter:=[Confidentiel]Info  
VARIABLE VERS BLOB ($vtCrypter;$vbCrypté)  
DOCUMENT VERS BLOB("cléPrivée.txt"; $vbcléPrivée)
```

```

Si (OK=1)
  CRYPTER BLOB ($vbCrypté; $vbcléPrivée)
  BLOB VERS DOCUMENT ("MiseAJour.txt"; $vbCrypté)
Fin de si

```

4) Le fichier de mise à jour peut alors être envoyé aux filiales (même en passant par un canal non sécurisé comme Internet). Si un tiers intercepte le fichier crypté, il sera dans l'incapacité de le décrypter sans la clé publique.

5) Chaque filiale peut, quant à elle, décrypter le document à l'aide de la clé publique :

```

  `Méthode DECRYPTER_INFOS
C_BLOB($vbCrypté; $vbcléPublique)
C_TEXTE($vtDécrypté)
C_HEURE ($vhRefDoc)

```

```

ALERTE ("Veuillez sélectionner le document crypté.")
$vhRefDoc:=Ouvrir document("") `Sélection du fichier MiseAJour.txt
Si (OK=1)

```

```

  FERMER DOCUMENT($vhRefDoc)
  DOCUMENT VERS BLOB(Document; $vbCrypté)
  DOCUMENT VERS BLOB("cléPublique.txt"; $vbcléPublique)
  Si (OK=1)
    DECRYPTER BLOB ($vbCrypté; $vbcléPublique)
    BLOB VERS VARIABLE($vbCrypté; $vtDécrypté)
    CREER ENREGISTREMENT ([Confidentiel])
    [Confidentiel]Info:=$vtDécrypté
    STOCKER ENREGISTREMENT([Confidentiel])
  Fin de si

```

```

  Fin de si

```

▼ Exemple 2 : Utilisation de deux clés.

Une société souhaite utiliser un système d'échange de données via Internet dans lequel chaque filiale reçoit des informations confidentielles mais envoie également ses propres informations à la maison-mère. Ce système a donc les impératifs suivants :

- Seul le destinataire doit pouvoir lire un message,
- On doit avoir la garantie que le message provient bien de l'expéditeur.

1) La maison-mère ainsi que chaque filiale génèrent leurs propres paires de clés (à l'aide de la méthode *GENERE_CLES_TXT*).

2) Chacune garde sa clé privée. Chaque filiale envoie sa clé publique à la maison-mère, qui elle-même envoie sa clé publique à chaque filiale. Cette transmission ne doit pas nécessairement être effectuée par un canal protégé, car la seule détention de la clé publique dans ce cas sera insuffisante pour décrypter une information.

3) Pour crypter une information à envoyer, une filiale ou la maison-mère exécute la méthode *CRYPTER_INFOS_2* qui utilise la clé privée de l'émetteur et la clé publique du destinataire pour crypter les données :

```

`Méthode CRYPTER_INFOS_2
C_BLOB($vbCrypté;$vbcléPrivée;$vbcléPublique)
C_TEXTE($vtCrypter)
C_HEURE ($vhRefDoc)

$vtCrypter:=[Confidentiel]Info
VARIABLE VERS BLOB ($vtCrypter;$vbCrypté)
` On charge sa propre clé privée...
DOCUMENT VERS BLOB("cléPrivée.txt"; $vbcléPrivée)
Si (OK=1)
    `...et la clé publique du récepteur
    ALERTE ("Veuillez sélectionner la clé publique du destinataire.")
    $vhRefDoc:=Ouvrir document("") `Sélection de la clé publique à charger
    Si (OK=1)
        FERMER DOCUMENT($vhRefDoc)
        DOCUMENT VERS BLOB(Document;$vbcléPublique)
        `Cryptage du BLOB avec les deux clés en paramètres
        CRYPTER BLOB ($vbCrypté; $vbcléPrivée; $vbcléPublique)
        BLOB VERS DOCUMENT ("MiseAJour.txt";$vbCrypté)
    Fin de si
Fin de si

```

4) Le fichier crypté peut alors être envoyé au destinataire via Internet. Si un tiers l'intercepte, il sera dans l'incapacité de le décrypter, même en connaissant les clés publiques, car il lui manquera la clé privée du destinataire.

5) Chaque destinataire peut, quant à lui, décrypter le document reçu, en utilisant sa clé privée et la clé publique de l'émetteur :

```

`Méthode DECRYPTER_INFOS_2
C_BLOB($vbCrypté;$vbcléPublique;$vbcléPrivée)
C_TEXTE($vtDécrypté)
C_HEURE ($vhRefDoc)

ALERTE ("Veuillez sélectionner le document crypté.")
$vhRefDoc:=Ouvrir document("") `Sélection du fichier MiseAJour.txt
Si (OK=1)
  FERMER DOCUMENT($vhRefDoc)
  DOCUMENT VERS BLOB(Document;$vbCrypté)
  `On charge sa propre clé privée
  DOCUMENT VERS BLOB("cléPrivée.txt"; $vbcléPrivée)
  Si (OK=1)
    `...et la clé publique de l'émetteur
    ALERTE ("Veuillez sélectionner la clé publique de l'envoyeur.")
    $vhRefDoc:=Ouvrir document("") `Sélection de la clé publique
    Si (OK=1)
      FERMER DOCUMENT($vhRefDoc)
      DOCUMENT VERS BLOB(Document;$vbcléPublique)
      `Décryptage du BLOB avec les deux clés en paramètres
      DECRYPTER BLOB ($vbCrypté; $vbcléPublique;$vbcléPrivée)
      BLOB VERS VARIABLE($vbCrypté; $vtDécrypté)
      CREER ENREGISTREMENT ([Confidentiel])
      [Confidentiel]Info:=$vtDécrypté
      STOCKER ENREGISTREMENT([Confidentiel])
    Fin de si
  Fin de si
Fin de si

```

Référence : GENERER CLES CRYPTAGE, DECRYPTER BLOB.

DECRYPTER BLOB

DECRYPTER BLOB (aD crypter; cl PubEmetteur{; cl PrivR cepteur})

Param�tres	Type	Description
aD�crypter	BLOB	→ Donn�es � d�crypter ← Donn�es d�crypt�es
cl�PubEmetteur	BLOB	→ Cl� publique de l�metteur
cl�PrivR�cepteur	BLOB	→ Cl� priv�e du r�cepteur

Cette commande permet de d crypter le contenu du BLOB aD crypter   l aide de la cl  publique de l metteur cl PubEmetteur ainsi que, optionnellement, de la cl  priv e du r cepteur cl PrivR cepteur.

Vous passez dans le param tre cl PubEmetteur le BLOB contenant la cl  publique de l metteur. Cette cl  a  t  g n r e par l metteur (  l aide de la commande GENERER CLES CRYPTAGE), qu il doit ensuite transmettre au r cepteur.

Le param tre optionnel cl PrivR cepteur doit recevoir la cl  priv e du r cepteur. Dans ce cas, le r cepteur doit  galement avoir g n r  une paire de cl s de cryptage   l aide de GENERER CLES CRYPTAGE et transmis sa cl  publique   l metteur. Le syst me de cryptage   deux cl s permet de garantir que seul l metteur peut avoir crypt  le message et seul le r cepteur peut le d crypter.

Pour plus d informations sur le syst me de cryptage   deux cl s, reportez-vous   la description de la [routine CRYPTER BLOB](#), page 32.

La commande DECRYPTER BLOB comporte une fonction de v rification d int grit  (*checksum*), afin d emp cher toute modification malveillante ou accidentelle du contenu du BLOB. Si le BLOB crypt  est endommag  ou modifi , la commande ne fera rien et retournera une erreur.

▼ Reportez-vous aux exemples de la commande CRYPTER BLOB.

R f rence : [GENERER CLES CRYPTAGE](#), [CRYPTER BLOB](#).

Définition structure Les commandes de ce thème ont été enrichies de manière à ce que le développeur puisse obtenir par programmation la totalité des propriétés des champs et des tables. A noter également la modification de la commande LIRE PROPRIETES CHAMP (cf. page 82).

**LIRE PROPRIETES
SAISIE CHAMP**

LIRE PROPRIETES SAISIE CHAMP (chpPtr | tableNum{; champNum}; nomEnum; obligatoire; nonSaisissable; nonModifiable)

Paramètres	Type	Description
chpPtr tableNum	Pointeur Numérique	→ Pointeur de champ ou Numéro de table
champNum	Numérique	→ Numéro de champ si un numéro de table est passé en premier paramètre
nomEnum	Alpha	← Nom de l'énumération associée ou Chaîne vide
obligatoire	Booléen	← Vrai = Obligatoire, Faux = Facultatif
nonSaisissable	Booléen	← Vrai = Non saisissable, Faux = Saisissable
nonModifiable	Booléen	← Vrai = Non modifiable, Faux = Modifiable

Cette commande retourne les propriétés relatives à la saisie de données du champ désigné par tableNum et champNum ou par chpPtr.

Vous pouvez passer :

- soit des numéros de table et de champ dans tableNum et champNum,
- soit un pointeur vers le champ dans chpPtr.

Note Les propriétés retournées par cette commande sont celles qui ont été définies au niveau de la fenêtre de structure de la base. Des propriétés similaires peuvent également être définies au niveau des formulaires.

Après l'exécution de la commande :

- Le paramètre nomEnum contient le nom de l'énumération associée au champ, s'il y en a une. Il est possible d'associer un énumération aux champs de type Alpha, Texte, Numérique, Entier, Entier long, Date, Heure et Booléen.
Si aucune énumération n'est associée au champ, ou si son type n'admet pas l'association d'énumération, une chaîne vide ("") est retournée.

- Le paramètre obligatoire retourne Vrai si le champ dispose de l'attribut "Obligatoire", Faux sinon. L'attribut Obligatoire peut être associé aux champs de tous types, hormis sous-table et BLOB.
- Le paramètre nonSaisissable retourne Vrai si le champ dispose de l'attribut "Non saisissable", Faux sinon. Un champ non saisissable ne peut qu'être lu, il n'accepte aucune saisie de données. L'attribut Non saisissable peut être associé aux champs de tous types, hormis sous-table et BLOB.
- Le paramètre nonModifiable retourne Vrai si le champ dispose de l'attribut "Non modifiable", Faux sinon. Un champ Non modifiable n'accepte qu'une seule saisie, et ne peut plus être modifié par la suite. L'attribut Non modifiable peut être associé aux champs de tous types, hormis sous-table et BLOB.

Référence : LIRE PROPRIETES CHAMP, LIRE PROPRIETES LIEN, LIRE PROPRIETES TABLE

LIRE PROPRIETES
LIEN

LIRE PROPRIETES LIEN (chpPtr | tableNum{; champNum}; tableDest;
champDest{; discriminant{; allerAuto{; retourAuto{}}})

Paramètres	Type		Description
chpPtr tableNum	Pointeur Num	→	Pointeur de champ ou Numéro de table
champNum	Numérique	→	Numéro de champ si un numéro de table est passé en premier paramètre
tableDest	Numérique	←	Numéro de la table cible ou 0 si aucun lien ne part du champ
champDest	Numérique	←	Numéro du champ cible ou 0 si aucun lien ne part du champ
discriminant	Numérique	←	Numéro du champ discriminant 0 si aucun champ discriminant
allerAuto	Booléen	←	Vrai = Lien aller automatique, Faux = Lien aller manuel
retourAuto	Booléen	←	Vrai = Lien retour automatique, Faux = Lien retour manuel

Cette commande retourne les propriétés du lien, s'il y en a un, qui part du champ source, désigné par tableNum et champNum ou par chpPtr.

Vous pouvez passer :

- soit des numéros de table et de champ dans tableNum et champ-Num,
- soit un pointeur vers le champ dans chpPtr.

Après l'exécution de la commande :

- Les paramètres tableDest et champDest contiennent respectivement le numéro de la table et du champ vers lesquels pointe le lien partant du champ source. Si aucun lien ne part du champ, ces paramètres contiennent 0.
- Le paramètre discriminant contient le numéro du champ discriminant (appartenant à la table cible) défini pour le lien. Si aucun champ discriminant n'a été défini pour le lien ou si aucun lien ne part du champ source, ce paramètre contient 0.
- Les paramètres allerAuto et retourAuto retournent Vrai si respectivement les options "Lien aller auto" et "Lien retour auto" ont été cochées pour le lien, Faux sinon.

Note Les deux derniers paramètres retournent également Vrai si aucun lien ne part du champ source (dans ce cas, leur valeur est non significative). La valeur des paramètres tableDest et champDest vous permet de vous assurer de l'existence d'un lien.

Référence : [LIRE PROPRIETES CHAMP](#), [LIRE PROPRIETES SAISIE CHAMP](#), [LIRE PROPRIETES TABLE](#)

LIRE PROPRIETES
TABLE

LIRE PROPRIETES TABLE (tablePtr | tableNum; invisible{; trigSvgdeNouv{; trigSvgdeEnr{; trigSupprEnr{; trigChargEnr}{})

Paramètres	Type	Description
tablePtr	Pointeur	→ Pointeur de table ou
tableNum	Num	Numéro de table
invisible	Booléen	← Vrai = Invisible, Faux = Visible
trigSvgdeNouv	Booléen	← Vrai = Trigger “Sur sauvegarde nouvel enreg” activé, sinon Faux
trigSvgdeEnr	Booléen	← Vrai = Trigger “Sur sauvegarde enregistrement” activé, sinon Faux
trigSupprEnr	Booléen	← Vrai = Trigger “Sur suppression enreg” activé, sinon Faux
trigChargEnr	Booléen	← Vrai = Trigger “Sur chargement enregistrement” activé, sinon Faux

Cette commande retourne les propriétés de la table désignée par tablePtr ou tableNum. Vous pouvez passer dans le premier paramètre soit un pointeur vers la table, soit le numéro de la table.

Après l’exécution de la commande :

- Le paramètre invisible retourne Vrai si la table dispose de l’attribut “Invisible”, Faux sinon. L’attribut Invisible permet de masquer la table dans les éditeurs standard de 4D (étiquettes, graphes...).
- Le paramètre trigSvgdeNouv retourne Vrai si le trigger “Sur sauvegarde nouvel enreg” a été activé pour la table, Faux sinon.
- Le paramètre trigSvgdeEnr retourne Vrai si le trigger “Sur sauvegarde enregistrement” a été activé pour la table, Faux sinon.
- Le paramètre trigSupprEnr retourne Vrai si le trigger “Sur suppression enreg” a été activé pour la table, Faux sinon.
- Le paramètre trigChargEnr retourne Vrai si le trigger “Sur chargement enregistrement” a été activé pour la table, Faux sinon.

Référence : [LIRE PROPRIETES CHAMP](#), [LIRE PROPRIETES SAISIE CHAMP](#), [LIRE PROPRIETES LIEN](#)

Documents système

CREER ALIAS

CREER ALIAS (cheminCible; cheminAlias)

Paramètre	Type	Description
cheminCible	Alpha	→ Nom ou chemin d'accès de la cible de l'alias/du raccourci
cheminAlias	Alpha	→ Nom ou chemin d'accès complet de l'alias/du raccourci à créer

La commande CREER ALIAS crée un alias (appelé “raccourci” sous Windows) du fichier ou dossier cible désigné par le paramètre cheminCible, avec le nom et l'emplacement définis dans le paramètre cheminAlias.

Vous pouvez créer un alias de tout type de document ou de dossier. L'icône de l'alias sera identique à celle de l'élément cible. Elle comportera en outre une petite flèche en bas à gauche et, sous MacOS, le libellé de l'alias apparaîtra en caractères italiques.

La commande n'affecte pas de libellé par défaut à l'alias, vous devez passer un nom dans le paramètre cheminAlias. Si vous passez uniquement un nom dans ce paramètre, l'alias est créé dans le dossier actif courant (généralement, le dossier contenant le fichier de structure de la base).

Note Sous Windows, les raccourcis sont des fichiers dont l'extension est “.LNK”. Si vous ne passez pas cette extension, la commande l'ajoute automatiquement.

Si vous passez une chaîne vide dans cheminCible, la commande ne fait rien.

- ▼ Votre base génère des fichiers texte intitulés “RapportNuméro”, stockés dans le dossier de la base. Vous souhaitez permettre à l'utilisateur de créer des raccourcis vers ces rapports et de les stocker où il le souhaite :

```
`Méthode CREER_RAPPORT
C_TEXTE($vtRapport)
C_ALPHA(250;$vtChemin)
C_ALPHA(80;$vaNom)
C_HEURE(vDoc)
C_ENTIER($NumRapport)
```

```
$vTRapport:=... `Edition du rapport
$NumRapport:=... `Calcul du numéro du rapport
$vaNom:="Rapport"+Chaine($NumRapport)+".txt" `Nom du fichier
vDoc:=Creer document($vaNom)
Si (OK=1)
  ENVOYER PAQUET(vDoc;$vTRapport)
  FERMER DOCUMENT(vDoc)
  `Ajout de l'alias
  CONFIRMER("Créer un alias pour ce rapport ?")
  Si (OK=1)
    $vtChemin:=Selectionner dossier ("Où souhaitez-vous créer l'alias ?")
    Si (OK=1)
      CREER ALIAS ($vaNom;$vtChemin+$vaNom)
    Fin de si
  Fin de si
Fin de si
```

Référence : RESOUDRE ALIAS.

RESOUDRE ALIAS

RESOUDRE ALIAS (cheminAlias; cheminCible)

Paramètre	Type	Description
cheminAlias	Alpha	→ Nom ou chemin d'accès complet de l'alias/du raccourci
cheminCible	Alpha	← Nom ou chemin d'accès complet de la cible de l'alias/du raccourci

La commande RESOUDRE ALIAS retourne le chemin d'accès complet du fichier ou dossier cible d'un alias (appelé "raccourci" sous Windows).

Vous passez dans cheminAlias le nom ou le chemin d'accès complet de l'alias.

Après l'exécution de la commande, la variable cheminCible contient le chemin d'accès complet du fichier ou dossier cible de l'alias.

Note Sous Windows, les raccourcis sont des fichiers dont l'extension est ".LNK". Si vous ne passez pas cette extension, la commande l'ajoute automatiquement.

Référence : CREER ALIAS.

SUPPRIMER
DOSSIER

SUPPRIMER DOSSIER (dossier)

Paramètre	Type	Description
dossier	Alpha	→ Nom ou chemin d'accès complet du dossier à supprimer

La commande SUPPRIMER DOSSIER supprime le dossier dont vous avez passé le nom ou le chemin d'accès complet dans dossier.

Seuls les dossiers vides peuvent être supprimés par cette commande.

- Si vous tentez de supprimer un dossier contenant des éléments, l'erreur -47 (Tentative de suppression d'un dossier non vide) est générée.
- Si vous passez dans dossier le chemin d'accès d'un fichier, ou une chaîne vide, ou encore le chemin d'accès d'un dossier inexistant, la commande ne fait rien et génère l'erreur -43 (Fichier non trouvé).

Vous pouvez intercepter ces erreurs à l'aide d'une méthode installée par la commande APPELER SUR ERREUR.

Référence : SUPPRIMER DOCUMENT.

LIRE ICONE
DOCUMENT

LIRE ICONE DOCUMENT(cheminAccès; icône{; taille})

Paramètres	Type	Description
cheminAccès	Alpha	→ Nom ou chemin d'accès du fichier duquel obtenir l'icône ou chaîne vide pour afficher la boîte de dialogue standard d'ouverture de fichiers
icône	Image	→ Champ ou variable image ← Icône du fichier
taille	Entier long	→ Taille de l'icône (en pixels)

La commande LIRE ICONE DOCUMENT retourne dans le champ ou la variable image 4D icône, l'icône du fichier dont vous avez passé le nom ou le chemin d'accès complet dans cheminAccès. Le fichier peut être de tout type (document, exécutable, raccourci ou alias...). Cependant, la commande ne retourne pas les icônes de dossiers.

Passez dans cheminAccès le chemin d'accès absolu du fichier dont vous souhaitez récupérer l'icône. Vous pouvez passer uniquement le nom du fichier ou un chemin d'accès relatif, dans ce cas le fichier doit se trouver dans le dossier courant de la base (généralement, le dossier contenant le fichier de structure de la base).

Si vous passez une chaîne vide dans cheminAccès, la boîte de dialogue standard d'ouverture de fichiers apparaît, permettant à l'utilisateur de désigner un fichier. Une fois la boîte de dialogue validée, la variable système *Document* contient le chemin d'accès complet du fichier sélectionné.

Passez dans le paramètre icône un champ ou une variable image 4D. Après l'exécution de la commande, ce paramètre contient l'icône du fichier (au format PICT).

Le paramètre optionnel taille vous permet d'indiquer les dimensions de l'image que vous souhaitez obtenir. La valeur du paramètre correspond à la longueur d'un côté du carré dans lequel l'image sera incluse. Généralement, les icônes sont définies en 32x32 pixels ("grande icône") ou 16x16 pixels ("petite icône"). Si vous passez 0 ou omettez le paramètre, la commande retourne l'icône dans sa plus grande taille disponible.

Environnement 4D

LIRE
INFORMATIONS
SERIALISATION

LIRE INFORMATIONS SERIALISATION(clé; utilisateur; société; connectés; maxUtilisateurs)

Paramètres	Type	Description
clé	Entier long ←	Clé unique du produit (crypté)
utilisateur	Alpha ←	Nom enregistré
société	Alpha ←	Organisation enregistrée
connectés	Entier long ←	Nombre d'utilisateurs connectés
maxUtilisateurs	Entier long ←	Nombre maximum d'utilisateurs

La commande LIRE INFORMATIONS SERIALISATION retourne diverses informations relatives à la sérialisation de l'application 4D courante.

- clé : identifiant unique du produit installé. Ce numéro unique correspond à une seule application 4D (4D Server, 4^e Dimension, 4D Runtime, etc.) installée sur un seul poste. Bien entendu, ce numéro est crypté.
- utilisateur : Nom de l'utilisateur de l'application, tel qu'il a été saisi au moment de l'installation.
- société : Nom de la société ou de l'organisation à laquelle appartient l'utilisateur, tel qu'il a été saisi au moment de l'installation.

- **connectés** : Nombre d'utilisateurs connectés au moment de l'exécution de la commande.
- **maxUtilisateurs** : Nombre maximal d'utilisateurs pouvant se connecter simultanément.

Note Les deux derniers paramètres retournent toujours 1 pour les versions monopostes de 4D, sauf lorsqu'il s'agit de versions de démonstration, auquel cas ils retournent 0.

La commande LIRE INFORMATION SERIALISATION s'inscrit dans le schéma général de protection des composants proposé par 4D, à compter de la version 6.7 du programme (pour plus d'informations sur les *composants*, reportez-vous au [paragraphe "Utilisation des composants 4D"](#), page 27).

Les développeurs de composants peuvent, s'ils le souhaitent, lier chaque copie de leur produit à une seule application 4D installée, afin d'empêcher toute copie illicite.

Le principe de fonctionnement du système est le suivant : un utilisateur souhaitant acquérir un composant fournit au développeur sa clé unique — générée à l'aide de la commande LIRE INFORMATION SERIALISATION. Cette opération peut, par exemple, être effectuée par l'intermédiaire d'un formulaire "Bon de commande" intégré à la version de démonstration du composant. La clé générée est unique : il n'existe qu'une clé par application 4D installée.

Le développeur du composant peut alors générer son propre numéro de série, en combinant la clé et l'algorithme de cryptage de son choix. Le composant livré comportera une fonction permettant de tester si les informations retournées par LIRE INFORMATION SERIALISATION correspondent bien à ce numéro de série. Dans le cas contraire, le composant sera rendu inutilisable.

Note Les développeurs de plug-ins peuvent également bénéficier de ce système de protection. Pour plus d'informations, reportez-vous à la documentation de *4D Plugin SDK*.

Images

ECRIRE FICHER IMAGE

ECRIRE FICHER IMAGE (nomFichier; image{; format))

Paramètres	Type	Description
nomFichier	Alpha	→ Nom ou chemin d'accès complet du fichier à écrire, ou chaîne vide
image	Image	→ Champ ou variable image à écrire
format	Alpha (4)	→ Code QuickTime de format d'export d'image (4 caractères) PICT par défaut

Cette commande vous permet de sauvegarder dans un fichier sur disque l'image passée dans le paramètre image, au format défini par format.

Cette commande utilise sous MacOS et Windows les routines de conversion de QuickTime (version 4 minimum recommandée). Si QuickTime n'est pas installé, la commande crée par défaut un fichier au format PICT.

Vous pouvez passer dans nomFichier le chemin d'accès complet du fichier à créer, ou uniquement le nom du fichier — auquel cas le fichier sera créé à côté du fichier de structure de la base. Sous Windows, vous devez également passer l'extension du fichier à créer.

Si vous passez une chaîne vide ("") dans nomFichier, la boîte de dialogue standard d'enregistrement de fichier apparaît, permettant à l'utilisateur de désigner le nom, l'emplacement et le format du fichier à créer.

Passez dans image la variable ou le champ image contenant l'image à stocker sur le disque.

Le paramètre optionnel format vous permet de définir le format dans lequel l'image doit être sauvegardée. Ce paramètre doit comporter 4 caractères, correspondant à un code QuickTime. Vous pouvez obtenir la liste des formats disponibles à l'aide de la commande LISTE TYPES IMAGES. Pour une description des codes de format standard QuickTime 4, reportez-vous au [paragraphe “Codes de conversion QuickTime 4”, page 51](#).

Si vous ne passez pas le paramètre format ou si QuickTime n'est pas installé, le fichier image est créé au format PICT.

Si l'exécution de la commande est correcte, la variable système *Document* contient le chemin d'accès complet du fichier créé et la variable système *OK* prend la valeur 1. En cas d'échec, *OK* prend la valeur 0.

Référence : LIRE FICHER IMAGE, LISTE TYPES IMAGES

LIRE FICHER IMAGE LIRE FICHER IMAGE (nomFichier; image)

Paramètres	Type	Description
nomFichier	Alpha	→ Nom ou chemin d'accès complet du fichier à lire, ou chaîne vide
image	Image	← Champ ou variable recevant l'image

Cette commande vous permet d'ouvrir l'image stockée dans le fichier disque désigné par nomFichier et de la placer dans le champ ou la variable 4D image.

Cette commande utilise sous MacOS et Windows les routines de conversion de QuickTime (version 4 minimum recommandée). Si QuickTime n'est pas installé, la commande ne peut ouvrir que les fichiers au format PICT.

Vous pouvez passer dans nomFichier le chemin d'accès complet du fichier à lire, ou uniquement le nom du fichier — auquel cas il doit se trouver à côté du fichier de structure de la base. Sous Windows, vous devez également passer l'extension du fichier.

Si vous passez une chaîne vide (""), la boîte de dialogue standard d'ouverture de documents apparaît, permettant à l'utilisateur de sélectionner le fichier à lire, ainsi que les formats disponibles (fournis par QuickTime 4).

Vous pouvez obtenir la liste des formats disponibles à l'aide de la commande LISTE TYPES IMAGES. Pour une description complète des codes de format standard QuickTime 4, reportez-vous au [paragraphe "Codes de conversion QuickTime 4", page 51](#).

Passez dans image la variable ou le champ image devant recevoir l'image lue.

Note Le format interne de l'image est conservé par QuickTime au sein de la variable ou du champ 4D. Par conséquent, il sera nécessaire de disposer de QuickTime pour relire l'image dans 4D.

Si l'exécution de la commande est correcte, la variable système *Document* contient le chemin d'accès complet du fichier ouvert et la variable système *OK* prend la valeur 1. En cas d'échec, *OK* prend la valeur 0.

Référence : ECRIRE FICHIER IMAGE, LISTE TYPES IMAGES

LISTE TYPES IMAGES LISTE TYPES IMAGES (tabFormats{; tabNoms})

Paramètres	Type	Description
tabFormats	Tableau Alpha (4) ←	Codes QuickTime des formats d'import/export disponibles
tabNoms	Tableau Alpha ←	Noms des formats

Cette commande remplit le tableau tabFormats avec les codes QuickTime d'import/export d'images disponibles sur la machine où elle est exécutée.

Le tableau tabNoms, optionnel, permet de récupérer le nom de chaque format d'image. Les noms des formats sont plus explicites que leurs codes.

Bien entendu, cette commande requiert que QuickTime (version 4 minimum) soit installé sur la machine. Dans le cas contraire, tabFormats contient uniquement le format PICT.

LISTE TYPES IMAGES peut être utilisée pour vérifier la présence des formats d'images requis pour des bases. Cette fonction s'avère particulièrement utile lorsqu'une base emploie des formats personnalisés, non installés par défaut (possibilité offerte par QuickTime 4).

Les informations recueillies dans le tableau tabNoms permettent en outre de construire et d'afficher un pop up menu contenant les formats d'export d'images disponibles.

**Codes de conversion
QuickTime 4**

Voici la liste des codes de conversion fournis en standard par QuickTime 4. Chaque code est formé impérativement de 4 caractères. A noter que cette liste peut varier en fonction des machines, QuickTime 4 permettant l'ajout de routines de conversion personnalisées.

Codes QuickTime 4 Noms

PICT	QuickDraw PICT (MacOS)
PICS	PICS
GIFf	GIF (Graphic Interchange Format)
PNGf	PNG (Portable Network Graphic)
TIFF	TIFF (Tagged Image File)
8BPS	Photoshop (2.5 et 3.0)
.SGI	Silicon Graphics
BMPf	BMP (Bitmap)
JPEG	JPEG (Joint Photographic Experts Group)
JPEG	JFIF
PNTG	MacPaint
TPIC	TGA (Targa)
qdgx	QuickDraw GX Picture (si QuickDraw GX installé)
qtif	QuickTime Image
FPix	FlashPix

Ces codes QuickTime d'import/export d'images sont utilisés par les commandes ECRIRE FICHIER IMAGE et LIRE FICHIER IMAGE.

Référence : [LIRE FICHIER IMAGE](#), [ECRIRE FICHIER IMAGE](#)

IMAGE VERS BLOB

IMAGE VERS BLOB (image; blobImage; format)

Paramètres	Type	Description
image	Image	→ Champ ou variable image
blobImage	BLOB	← BLOB devant contenir l'image
format	Alpha (4)	→ Format d'image (4 caractères)

Cette commande convertit une image stockée dans une variable ou un champ 4D dans un autre format, et place l'image résultante dans un BLOB.

Vous passez dans le paramètre image une variable ou un champ 4D de type image et dans le paramètre blobImage la variable ou le champ BLOB devant contenir l'image convertie.

Vous passez dans le paramètre format une chaîne de 4 caractères indiquant le format de conversion souhaité.

Ce format peut être soit un code QuickTime (cf. [paragraphe “Codes de conversion QuickTime 4”, page 51](#)), auquel cas QuickTime version 4 minimum devra être installé sur le poste, soit GIFF (format GIF) ou WBMP (Wireless Bitmap), ces deux derniers codes ne nécessitant pas la présence de QuickTime.

Après l'exécution de la commande, blobImage contient l'image au format souhaité.

Si la conversion s'est déroulée correctement, la variable système *OK* prend la valeur 1. Si la conversion échoue (absence de QuickTime 4 ou convertisseur non disponible), *OK* prend la valeur 0 et le BLOB est généré vide (0 octet).

Référence : BLOB VERS IMAGE, ECRIRE FICHIER IMAGE, LISTE TYPES IMAGES, IMAGE VERS GIF.

BLOB VERS IMAGE

BLOB VERS IMAGE (blobImage; image)

Paramètres	Type	Description
blobImage	BLOB	→ BLOB contenant une image
image	Image	← Champ ou variable image

Cette commande place dans un champ ou une variable image 4D une image stockée dans un BLOB, quel que soit son format initial (compatible QuickTime 4).

Cette commande nécessite sous MacOS et Windows la présence de QuickTime version 4 minimum. Si QuickTime 4 n'est pas installé, la commande ne fait rien.

Le fonctionnement de cette commande est analogue à celui de la commande LIRE FICHIER IMAGE ; elle s'applique simplement à un BLOB et non à un fichier. Elle permet d'afficher à tout moment des images stockées en format natif dans des BLOBs à l'aide, par exemple, de la commande DOCUMENT VERS BLOB ou IMAGE VERS BLOB.

Vous passez dans le paramètre blobImage le BLOB contenant l'image. L'image peut être de tout format compatible QuickTime (version 4 minimum). Vous pouvez obtenir la liste des formats disponibles à l'aide de la commande LISTE TYPES IMAGES. Pour une description complète des codes de format standard QuickTime 4, reportez-vous au [paragraphe "Codes de conversion QuickTime 4", page 51](#).

Vous passez dans le paramètre image la variable ou le champ 4D de type image devant afficher l'image.

Note Le format interne de l'image est conservé par QuickTime au sein de la variable ou du champ 4D. Par conséquent, il sera nécessaire de disposer de QuickTime pour afficher l'image dans 4D.

Après l'exécution de la commande, image contient l'image affichable dans 4D.

Si la commande a été exécutée correctement, la variable système OK prend la valeur 1. En cas d'échec (absence de QuickTime 4, BLOB ne contenant pas d'image, format d'image inconnu...), OK prend la valeur 0 et le champ ou la variable image 4D est vide.

Référence : [ECRIRE FICHIER IMAGE](#), [LISTE TYPES IMAGES](#), [IMAGE VERS BLOB](#).

CREER IMAGETTE

CREER IMAGETTE (source; dest{; largeur{; hauteur{; mode{; profondeur}}))

Paramètres	Type	Description
source	Image	→ Champ ou variable image 4D à passer en imagette
dest	Image	← Imagette résultante
largeur	Entier	→ Largeur de l’imagette en pixels Par défaut = 48
hauteur	Entier	→ Hauteur de l’imagette en pixels Par défaut = 48
mode	Entier	→ Mode de création de l’imagette Par défaut = proportionnelle centrée (6)
profondeur	Entier	→ Nombre de couleurs de l’imagette en bits/pixel Par défaut = profondeur écran courante (0)

Cette commande retourne une imagette à partir d’une image source. Les imagettes sont généralement utilisées pour la prévisualisation d’images dans le cadre d’applications multimédia ou de sites Web.

Note Cette commande ne nécessite pas la présence de QuickTime.

Passez dans source la variable ou le champ image 4D contenant l’image source à réduire sous forme d’imagette, et dans dest la variable ou le champ image 4D devant recevoir l’imagette résultante.

Les paramètres optionnels largeur et hauteur vous permettent de définir la taille en pixels de l’imagette que vous souhaitez obtenir. Si vous omettez ces paramètres, la taille par défaut de l’imagette sera de 48X48 pixels.

Le paramètre optionnel mode vous permet de définir le mode de création de l’imagette, c’est-à-dire la manière dont elle sera réduite. Vous pouvez utiliser l’un des trois modes suivants, accessibles par l’intermédiaire de constantes prédéfinies disponibles dans le thème “Formats d’affichage des images” :

Constante	Type	Valeur
Non tronquée	Entier long	2
Proportionnelle	Entier long	5
Proportionnelle centrée	Entier long	6 (défaut)

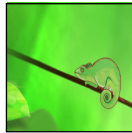
Note Seules ces trois constantes peuvent être utilisées avec CREER IMAGETTE. Les autres constantes du thème “Formats d’affichage des images” ne s’appliquent pas à cette commande.

Si vous omettez ce paramètre, le mode 6 (Proportionnelle centrée) est appliqué par défaut. Le résultat des différents modes est illustré ci-dessous :

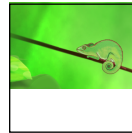
Image originale



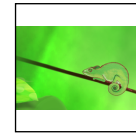
Imagettes résultantes (48 X 48)



Modes Non tronquée = 2



Proportionnelle = 5



Proportionnelle centrée = 6
(mode par défaut)

Note Avec les modes “Proportionnelle” et “Proportionnelle centrée”, les espaces vides apparaîtront blancs dans les imagettes — lorsque ces modes sont appliqués aux champs ou variables images dans les formulaires 4D, les espaces vides sont transparents.

Le paramètre optionnel profondeur vous permet de définir le nombre de couleurs (c’est-à-dire la profondeur d’écran) à conserver dans l’image de destination. Vous devez passer dans ce paramètre un entier correspondant au nombre de bits par pixel : 1, 2, 4, 8, 16 ou 32. Passez 0 pour utiliser la profondeur écran courante (valeur par défaut).

Langage

Nom methode courante

Nom methode courante → Alpha

Paramètres	Type	Description
------------	------	-------------

Résultat	Alpha	← Nom de la méthode d'appel
----------	-------	-----------------------------

La commande Nom methode courante retourne le nom de la méthode dans laquelle elle est appelée. Cette commande est utile dans le cadre du débogage de méthodes génériques.

Suivant le type de méthode d'appel, la chaîne retournée peut prendre l'une des formes suivantes :

Méthode d'appel	Chaîne retournée
Méthode base	NomMéthode
Trigger	Trigger sur [NomTable]
Méthode projet	NomMéthode
Méthode formulaire	[NomTable]NomFormulaire
Méthode objet	[NomTable]NomFormulaire.NomObjet

Cette commande ne doit pas être appelée depuis une formule 4D.

Protocole sécurisé Ce nouveau thème regroupe les commandes de cryptage des données utilisées par le protocole de connexion sécurisée SSL (*Secured Socket Layer*) pouvant désormais être utilisé par le serveur Web 4D. Pour une description détaillée du protocole SSL, reportez-vous au [paragraphe “Connexions sécurisées via SSL”](#), page 111.

**GENERER CLES
CRYPTAGE**

GENERER CLES CRYPTAGE (cléPrivée; cléPublique{; longueur})

Paramètres	Type	Description
cléPrivée	BLOB	→ BLOB devant recevoir la clé privée
cléPublique	BLOB	→ BLOB devant recevoir la clé publique
longueur	Entier long	→ Longueur des clés en bits [386...1024] Par défaut = 512

Cette commande génère une nouvelle paire de clés RSA. Ces clés, destinées au cryptage/décryptage des données, sont à la base du système de sécurisation des données proposé par 4D 6.7. Elles pourront être utilisées, dans le cadre du protocole SSL, avec le serveur Web 4D (cryptage et sécurité des communications) mais également dans toute base de données (cryptage de données).

Après l’exécution de la commande, les BLOB passés dans les paramètres cléPrivée et cléPublique contiennent une nouvelle paire de clés de cryptage.

Le paramètre optionnel longueur vous permet de préciser la taille (en bits) des clés que vous souhaitez obtenir. Plus une clé est longue, plus son décryptage “frauduleux” sera difficile.
En contrepartie, plus les clés sont longues, plus les délais d’exécution ou de réponse seront importants, en particulier dans le cadre d’une connexion SSL.
Par défaut (si vous omettez le paramètre longueur), la taille des clés générée est de 512 bits, ce qui correspond au compromis sécurité/rapidité généralement admis. Pour augmenter la sécurité dans ce cas, vous pouvez envisager de changer de paire de clés assez fréquemment, par exemple tous les six mois.

Vous pouvez générer des clés de 1024 bits, ce qui renforce la sécurité du cryptage, mais ralentira les connexions de votre application Web.

- Note*
- Attention si vous générez des clés dans le but d'établir une demande de certificat SSL : seules les clés de 512 et de 1024 bits sont admises.
 - Les clés d'une longueur supérieure à 512 bits ne sont pas acceptées par une grande partie des navigateurs Web actuels.

Les clés générées par cette commande sont au format PEM (*Privacy Enhanced Mail*), ce qui signifie que leur contenu peut être copié et collé dans un eMail sans risque d'altération. Une fois que vous avez obtenu une paire de clés, vous pouvez générer un document texte (par exemple à l'aide de la commande BLOB VERS DOCUMENT) et stocker les clés dans un endroit sûr.

La clé privée ne doit jamais être diffusée, sous quelque forme que ce soit.

RSA, clés privées et clés publiques

L'algorithme de cryptage RSA employé par la commande GENERER CLES CRYPTAGE est basé sur un système de cryptage à double clé : une clé privée et une clé publique. Comme son nom l'indique, la clé publique peut être diffusée auprès de tiers, et permet le décryptage des informations. Il lui correspond une clé privée unique, utilisée pour crypter les données. La clé privée sert au cryptage ; la clé publique, au décryptage (ou inversement). Ce qui est crypté avec une clé ne peut être décrypté qu'avec l'autre.

Les fonctions de cryptage du protocole SSL sont basées sur ce principe, la clé publique étant incluse dans le certificat envoyé aux browsers (cf. [paragraphe "Connexions sécurisées via SSL", page 111](#)).

Ce mode de cryptage est également utilisé par la première syntaxe des commande CRYPTER BLOB et DECRYPTER BLOB. Ce principe requiert que la clé publique soit diffusée de manière confidentielle.

Il est possible de mêler les clés publiques et privées de deux intervenants pour crypter des données de telle manière que seul le récepteur peut décrypter les données, et seul l'émetteur peut les avoir cryptées. C'est le principe de la seconde syntaxe des commandes CRYPTER BLOB et DECRYPTER BLOB.

- ▼ Reportez-vous à l'exemple de la commande CRYPTER BLOB.

Référence : [CRYPTER BLOB](#), [GENERER DEMANDE CERTIFICAT](#)

**GENERER DEMANDE
CERTIFICAT**

GENERER DEMANDE CERTIFICAT(cléPrivée; demCertif; tabCodes; tabNoms)

Paramètres	Type	Description
cléPrivée	BLOB	→ BLOB contenant la clé privée
demCertif	BLOB	→ BLOB devant recevoir la demande de certificat
tabCodes	Tableau Entier long	→ Liste des codes d'informations
tabLibellés	Tableau Alpha	→ Liste des libellés d'informations

Cette commande permet de générer une demande de certificat au format PEM (*Privacy Enhanced Mail*), directement exploitable par des autorités de certification telles que Verisign[®] ou Thawthe[®]. Le certificat est une pièce essentielle du fonctionnement du protocole SSL dans le cadre d'un serveur Web. Il est envoyé à chaque browser se connectant en mode SSL. Il contient la "carte d'identité" du site Web (repreant les informations que vous saisissez dans la commande), ainsi que sa clé publique — permettant aux browsers de décrypter les informations reçues. En outre, le certificat contient diverses informations ajoutées par l'autorité de certification.

Note Pour plus d'informations sur le fonctionnement du protocole SSL avec le serveur Web 4D, reportez-vous au [paragraphe "Connexions sécurisées via SSL"](#), page 111.

La demande de certificat nécessite une paire de clés générée à l'aide de la commande [GENERER CLES CRYPTAGE](#) et contient diverses informations. C'est en combinant cette demande avec d'autres paramètres qui lui sont propres, que l'autorité de certification sera en mesure de générer un certificat.

Passez dans cléPrivée un BLOB contenant la clé privée générée avec la commande [GENERER CLES CRYPTAGE](#).

Passez dans demCertif un BLOB vide. Après l'exécution de la commande, il contiendra la demande de certificat au format PEM. Vous pouvez placer cette demande dans un fichier texte, par exemple à l'aide de la commande BLOB VERS DOCUMENT, pour la faire parvenir à l'autorité de certification.

La clé privée est utilisée pour générer la demande de certificat mais ne doit pas être envoyée à l'autorité de certification.

Vous devez remplir les tableaux `tabCodes` (de type entier long) et `tabLibellés` (de type alpha) avec, respectivement, les numéros de code et les libellés des informations destinées à l'autorité de certification.

Les codes et les libellés attendus peuvent varier en fonction de l'autorité de certification et du mode d'utilisation du certificat. Toutefois, dans le cadre d'une utilisation standard du certificat (connexions d'un serveur Web via SSL), les tableaux doivent contenir les éléments suivants :

Informations à fournir	tabCodes	tabLibellés (Exemples)
<i>CommonName</i> : Nom du domaine	13	www.4D.fr
<i>CountryName</i> : Code du pays (deux lettres)	14	FR
<i>LocalityName</i> : Ville	15	Clichy
<i>StateOrProvinceName</i> : Département, Etat...	16	Hauts de Seine
<i>OrganizationName</i> : Raison sociale	17	4D
<i>OrganizationUnit</i> : Service/Personne en charge du serveur	18	Web Administrator

L'ordre dans lequel les codes et les informations sont insérés dans les tableaux n'a pas d'importance, en revanche les deux tableaux doivent être "synchronisés" : si l'élément {3} du tableau `tabCodes` contient la valeur *15* (nom de la ville), l'élément {3} du tableau `tabLibellés` doit contenir cette information, dans notre exemple *Clichy*.

- ▼ Un formulaire "Demande de certificat" comporte les six champs nécessaires à l'établissement d'une demande de certificat standard. Le bouton **Générer** crée un document sur disque contenant la demande de certificat.

Le document "Cléprivée.txt" contient la clé privée (générée à l'aide la commande **GENERER CLES CRYPTAGE**) doit déjà être présent sur le disque.

` Méthode objet du bouton bGénérer

```
C_BLOB($vbcléPrivée;$vbDemandeCert)
```

```
C_ENTIER LONG($NumTable)
```

```
TABLEAU ENTIER LONG ($tLCodes;6)
```

```
TABLEAU ALPHA (80;$tAInfos;6)
```

```
$NumTable:=Table(Table du formulaire courant)
```

```
Boucle ($i;1;6)
```

```
  $tAInfos{$i}:= Champ($NumTable;$i)
```

```
  $tLCodes{$i}:= $i+12
```

```
Fin de boucle
```

```
Si (Chercher dans tableau($tAInfos;"") # -1)
```

```
  ALERTE ("Vous devez remplir tous les champs.")
```

```
Sinon
```

```
  ALERTE ("Sélectionnez votre clé privée.")
```

```
  $vhRefDoc:=Ouvrir document("")
```

```
  Si (OK=1)
```

```
    FERMER DOCUMENT($vhRefDoc)
```

```
    DOCUMENT VERS BLOB(Document;$vbcléPrivée)
```

```
  Fin de si
```

```
  GENERER DEMANDE CERTIFICAT($vbcléPrivée;$vbDemandeCert;  
                               $tLCodes; $tAInfos)
```

```
  BLOB VERS DOCUMENT ("Demande.txt";$vbDemandeCert)
```

```
Fin de si
```

Référence : **GENERER CLES CRYPTAGE**

Ressources

**Lire ID ressource
composant**

Lire ID ressource composant (nomComp; typeRes; numResOriginal) →
Numérique

Paramètres	Type	Description
nomComp	Alpha (32)	→ Nom du composant référençant la res- source
typeRes	Alpha (4)	→ Type de ressource (4 caractères) PICT ou STR#
numResOriginal	Numérique	→ Numéro original de la ressource, avant installation du composant
Résultat	Numérique	← Numéro courant de la ressource

La commande Lire ID ressource composant permet aux développeurs de composants de s’assurer que leurs appels de ressources personnalisées de type PICT ou STR# seront correctement effectués, même si les numéros de ces ressources ont été modifiés au moment de l’installation.

En effet, lorsqu’un composant utilisant des ressources est installé par 4D Insider, le programme peut renuméroter automatiquement les nouvelles ressources si des ressources de même numéro existent déjà dans la base.

Note Pour plus d’infomations sur les *composants* dans 4^e Dimension 6.7, reportez-vous au [paragraphe “Utilisation des composants 4D”](#), page 27, et à la documentation de 4D Insider.

La commande Lire ID ressource composant permet donc de connaître le numéro courant (réel) de chaque ressource utilisée par un composant, à partir de son type et de son numéro original.

Passez dans le paramètre nomComp le nom du composant utilisant la ressource.

Passez dans le paramètre typeRes le type de la ressource (impérativement composé de 4 caractères). La commande Lire ID ressource composant accepte uniquement les ressources de type PICT et STR#.

Note Les images stockées dans la bibliothèque d'images de 4D sont des *pseudo* ressources. Elles ne sont pas gérées par la commande Lire ID ressource composant.

Passez dans numResOriginal le numéro original de la ressource, c'est-à-dire celui défini au moment de la création du composant.

La fonction retourne alors le numéro attribué à la ressource dans l'application courante.

Si aucune ressource ne correspond à numResOriginal, Lire ID ressource composant retourne la valeur saisie dans numResOriginal.

- ▼ Le code ci-dessous ne permet pas de garantir que les appels de ressources seront correctement effectués :

Si les ressources sont renumérotées, cet appel sera incorrect

vNumRes := 15000

LISTE DE CHAINES VERS TABLEAU(vNumRes; tabChaines; fichierRes)

Il est fortement conseillé de préférer le code suivant :

Cet appel sera correct dans tous les cas

vNumRes := Lire ID ressource composant("MonComp"; "STR#"; 15000)

LISTE DE CHAINES VERS TABLEAU (vNumRes; tabChaines; fichierRes)

Serveur Web

Connexion Web sécurisée

Connexion Web sécurisée → Booléen

Paramètres	Type	Description
Résultat	Booléen ←	Vrai = la connexion Web est sécurisée Faux = la connexion Web n'est pas sécurisée

Cette fonction retourne un booléen indiquant si la connexion au serveur Web 4D s'effectue en mode sécurisé via SSL (la requête débute par "https:" au lieu de "http:").

- Si la connexion est effectuée en SSL, la fonction retourne Vrai.
- Si la connexion est effectuée en mode classique (non sécurisé), la fonction retourne Faux.

Note Pour plus d'informations le protocole SSL, reportez-vous au [paragraphe "Connexions sécurisées via SSL", page 111](#).

Cette commande permet par exemple, le cas échéant, de refuser les tentatives de connexion en mode non sécurisé. Pour plus d'informations sur ce point, reportez-vous au [paragraphe “Mise à jour des serveurs Web existants”](#), page 116.

Référence : GENERER DEMANDE CERTIFICAT

ENVOYER TEXTE HTML

ENVOYER TEXTE HTML (texteHTML{;sansContexte})

Paramètres	Type	Description
texteHTML	Texte	→ Champ ou variable texte au format HTML à envoyer au browser
sansContexte	Booléen	→ Vrai = Passage en mode sans contexte Faux ou omis = Conservation du mode en cours

Cette commande permet d'envoyer directement des données texte formatées en HTML.

Le paramètre texteHTML contient les données à envoyer. 4D n'effectue aucun contrôle sur le contenu de ce paramètre, vous devez donc veiller à ce que le codage HTML soit correct. Le texte doit être encodé en ISO Latin-1.

Note Cette commande équivaut strictement à l'envoi d'un BLOB ayant le type “text/html” à l'aide de la commande ENVOYER BLOB HTML.

Le paramètre sansContexte vous permet d'indiquer au serveur Web 4D que vous souhaitez passer du mode “contextuel” au mode “sans contexte” lors de l'exécution de cette commande.

Pour utiliser le mode sans contexte, passez Vrai dans sansContexte. Pour conserver le mode courant, passez Faux dans sansContexte ou omettez ce paramètre.

Les éventuelles références aux variables 4D et balises 4DSCRIPT dans le texte sont toujours analysées, quel que soit le mode.

▼ La méthode suivante :

```
TEXTE VERS BLOB("<html><head></head><body>" + Chaîne(Heure
courante) + "</body></html>";
$blob; Texte sans longueur)
ENVOYER BLOB HTML ($blob;"text/html")
```


... peut être remplacée par :

ENVOYER TEXTE HTML ("`<html><head></head><body>`" + **Chaîne(Heure courante)** + "`</body></html>`")

LIRE ENTETE HTTP

LIRE ENTETE HTTP (entete | tabChamps{; tabValeurs})

Paramètres	Type	Description
entete	Texte	← En-tête HTTP de la requête
tabChamps	tableau Texte	← Champs de l'en-tête HTTP
tabValeurs	tableau Texte	← Contenu des champs de l'en-tête HTTP

La commande LIRE ENTETE HTTP retourne, soit sous forme de chaîne, soit sous forme de deux tableaux, l'en-tête HTTP de la requête en cours de traitement.

Cette commande fonctionne uniquement en mode sans contexte. Elle peut être appelée depuis toute méthode (Méthode base Sur connexion Web, Méthode base Sur authentification Web, méthode appelée par '/4DACTION'...) exécutée dans un process Web en mode sans contexte. Appelée en mode contextuel, LIRE ENTETE HTTP retourne des chaînes vides.

- Première syntaxe : LIRE ENTETE HTTP (entete)
Lorsque vous utilisez cette syntaxe, le résultat retourné dans la variable entete est du type suivant :

"GET /page.html HTTP\1.0"+Caractere(13)+Caractere(10)+"User-Agent: browser"+Caractere(13)+Caractere(10)+"Cookie: C=HELLO"

Chaque champ d'en-tête est séparé par une séquence CR+LF (*Retour chariot+Retour à la ligne*), sous Windows et MacOS.

- Seconde syntaxe : LIRE ENTETE HTTP (tabChamps; tabValeurs)
Lorsque vous utilisez cette syntaxe, les résultats retournés dans les tableaux tabChamps et tabValeurs sont du type suivant :

tabChamps{1} = "X-METHOD"	tabValeurs{1} = "GET" *
tabChamps{2} = "X-URL"	tabValeurs{2} = "/page.html" *
tabChamps{3} = "X-VERSION"	tabValeurs{3} = "HTTP/1.0" *
tabChamps{4} = "User-Agent"	tabValeurs{4} = "browser"
tabChamps{5} = "Cookie"	tabValeurs{5} = "C=HELLO"

* Ces trois premiers éléments ne correspondent pas à des champs HTTP. Ils constituent la première ligne de la requête.

Conformément à la norme HTTP, les noms des champs sont toujours libellés en anglais.

Note La commande `FIXER ENTETE HTTP` accepte également deux syntaxes en version 6.7. Reportez-vous au [paragraphe “FIXER ENTETE HTTP \(Thème Serveur Web\)”](#), page 73.

A titre indicatif, voici une liste non exhaustive des champs HTTP pouvant être présents dans une requête :

- **Accept** : ce que le navigateur est susceptible d'accepter comme contenu.
 - **Accept-Language** : la ou les langue(s) acceptée(s) par le navigateur (pour information). Permet de choisir une page d'accueil en fonction de la langue préférée du navigateur.
 - **Cookie** : liste des cookies.
 - **From** : adresse e-mail de l'utilisateur du navigateur.
 - **Host** : nom ou adresse du serveur (par exemple, dans le cas de l'URL `http://monserveurweb/mapage.html`, Host prend la valeur "monserveurweb"). Permet de gérer les cas où plusieurs noms pointent vers la même adresse IP (*virtual hosting*).
 - **Referer** : provenance de la requête (par exemple `http://monserveurweb/mapage1.html`), c'est-à-dire la page que l'utilisateur affiche s'il clique sur le bouton Précédent de son navigateur.
 - **User-Agent** : nom et version du navigateur ou du proxy.
- ▼ Cette méthode permet de récupérer le contenu de tout champ d'en-tête de requête HTTP :

```
` Méthode projet GetHTTPField
` GetHTTPField ( Texte ) -> Texte
` GetHTTPField ( Nom en-tête HTTP ) -> Contenu en-tête HTTP
C_TEXTE($0;$1)
C_ENTIER LONG($vElem)
TABLEAU TEXTE($noms;0)
TABLEAU TEXTE($valeurs;0)
$0:=""
LIRE ENTETE HTTP ($noms;$valeurs)
$vElem:=Chercher dans tableau($noms;$1)
Si ($vElem>0)
    $0:=$valeurs{$vElem}
Fin de si
```

- Une fois que cette méthode projet est écrite, vous pouvez l'appeler ainsi :

```
` Contenu de l'en-tête Cookie
$cookie:=GetHTTPField("Cookie")
```

- Vous pouvez également envoyer des pages différentes en fonction de la langue du navigateur (par exemple dans la Méthode base Sur connexion Web) :

```
$langue:=GetHTTPField("Accept-Language")
```

Au cas ou

```
:( $langue="@fr@" ) ` Français (cf. liste ISO 639)
```

```
    ENVOYER FICHER HTML("index_fr.html")
```

```
:( $langue="@es@" ) ` Espagnol (cf. liste ISO 639)
```

```
    ENVOYER FICHER HTML("index_es.html")
```

Sinon

```
    ENVOYER FICHER HTML("index.html")
```

Fin de cas

- Exemple de gestion des hôtes virtuels (par exemple dans la Méthode base Sur connexion Web). Les trois noms "home_site.com", "home_site1.com" et "home_site2.com" pointent vers la même adresse IP, par exemple 192.1.2.3.

```
$host:=GetHTTPField("Host")
```

Au cas ou

```
:( $host="www.site1.com" )
```

```
    ENVOYER FICHER HTML("home_site1.com")
```

```
:( $host="www.site2.com" )
```

```
    ENVOYER FICHER HTML("home_site2.com")
```

Sinon

```
    ENVOYER FICHER HTML("home_site.com")
```

Fin de cas

Référence : FIXER ENTETE HTTP.

LIRE VARIABLES
FORMULAIRE WEB

LIRE VARIABLES FORMULAIRE WEB (tabNoms;tabValeurs)

Paramètres	Type	Description
tabNoms	tableau Texte ←	Noms des variables du formulaire Web
tabValeurs	tableau Texte ←	Valeurs des variables du formulaire Web

La commande LIRE VARIABLES FORMULAIRE WEB remplit les tableaux texte tabNoms et tabValeurs avec, respectivement, les noms et les valeurs des variables contenues dans un formulaire Web “soumis” (c’est-à-dire envoyé au serveur Web).

Cette commande récupère la valeur de toutes les variables pouvant être incluses dans des pages HTML : zones de saisie, boutons, cases à cocher, boutons radio, pop up menus, listes d’options...

Note Dans le cas des cases à cocher, le nom de la variable et sa valeur (“ON”) ne sont retournés que si la case est effectivement cochée.

La commande fonctionne en mode contextuel et en mode sans contexte, lorsqu’elle est appelée dans les conditions suivantes :

- un formulaire est soumis avec la méthode “POST” (action commençant par /4DACTION, /4DMETHOD ou /4DCGI),
- un formulaire est soumis avec la méthode “GET” (action commençant par /4DACTION, /4DMETHOD ou /4DCGI),
- un URL contenant une chaîne d’interrogation est envoyé au serveur Web.

Cette commande peut être appelée, selon les besoins, dans la Méthode base Sur connexion Web ou toute autre méthode 4D qui résulte de la soumission d’un formulaire.

Note Dans 4D version 6.5, lorsque le serveur Web 4D recevait un formulaire (soumis avec la méthode POST), il affectait les valeurs des zones de saisie du formulaire à des variables¹ texte 4D, si elles portaient le même nom. Ce fonctionnement pouvait entraîner des erreurs en mode compilé si les variables n’étaient pas déclarées. Désormais, avec 4D version 6.7, que l’application soit compilée ou non, si une variable n’est pas déclarée, aucune erreur ne sera générée. La variable sera simplement ignorée, de plus elle restera accessible grâce à la commande LIRE VARIABLES FORMULAIRE WEB.

1. Avec 4D version 6.7, ce mécanisme peut également s’appliquer à des éléments de tableau et à des champs.

- ▼ Un formulaire contient deux champs, *vNOM* et *vVILLE*, qui reçoivent les valeurs “MARTIN” et “PARIS”. L’action associée au formulaire est “/4DACTION/WEBFORM”.
- Si la méthode du formulaire est POST (cas le plus souvent utilisé), les données saisies ne seront pas visibles dans l’URL (c’est-à-dire `http://127.0.0.1/4DACTION/WEBFORM`).
- Si la méthode du formulaire est GET, les données seront visibles dans l’URL (c’est-à-dire `http://127.0.0.1/4DACTION/WEBFORM?vNOM=MARTIN&vVILLE=PARIS`).

La méthode *WEBFORM* peut être de la forme suivante :

TABLEAU TEXTE(\$tnoms;0)

TABLEAU TEXTE(\$tvaleurs;0)

LIRE VARIABLES FORMULAIRE WEB(\$tnoms;\$tvaleurs)

On obtient alors :

`$tnoms{1} = "vNOM"`

`$tnoms{2} = "vVILLE"`

`$tvaleurs{1} = "MARTIN"`

`$tvaleurs{2} = "PARIS"`

En outre, la variable *vNOM* contient MARTIN et la variable *vVILLE* contient PARIS.

Précisions sur les URL,
les formulaires Web et
les actions associées

Un URL est constitué des éléments suivants :

- protocole://adresse/chemin [#section], par exemple :
`http://127.0.0.1/chemin` ou `http://127.0.0.1/chemin#lasection`
- ou protocole://adresse/chemin [?interrogation], par exemple :
`http://127.0.0.1/chemin?var=valeur`.

Un formulaire est composé de “zones de saisie” (zones de texte, boutons, cases à cocher), chacune ayant un nom.

Lorsqu’un formulaire est “soumis” (une requête est envoyée au serveur Web), la requête comporte, entre autres, la liste des zones de saisie et leurs valeurs respectives.

Il y a deux “méthodes” pour soumettre un formulaire (4D accepte indifféremment l’une ou l’autre) :

- POST, généralement utilisée pour l’insertion de données dans le serveur Web — vers une base de données,
- GET, généralement utilisée pour l’interrogation du serveur Web — données en provenance d’une base de données.

Commandes modifiées

Cette section décrit les commandes existantes dont le nom, la syntaxe ou le comportement a été modifié(e) dans la version 6.7. Ces commandes sont classées par ordre alphabétique.
Les éventuels nouveaux paramètres sont inscrits en *caractères italiques*.

CHANGER PROPRIETES LISTE (Thème Listes hiérarchiques)

CHANGER PROPRIETES LISTE(liste; apparence{; icône{; ligneHauteur {;doubleClic}}})

Paramètres	Type	Description
liste	RefListe →	Numéro de référence de la liste
apparence	Numérique →	Style graphique de la liste
icône	Numérique →	Référence de ressource MacOS 'cicn' ou 0 = icône par défaut de la plate-forme
ligneHauteur	Numérique →	Hauteur minimale de la ligne (pixels)
<i>doubleClic</i>	Entier long →	Déploiement/contraction sur double-clic 0 = autoriser, 1= empêcher

La commande CHANGER PROPRIETES LISTE admet un nouveau paramètre optionnel, permettant d'empêcher que le double-clic sur un élément de la liste ne provoque le déploiement ou la contraction de sa sous-liste.

Par défaut, une sous-liste est déployée ou contractée en cas de double-clic sur l'élément parent. Certains types d'interfaces peuvent toutefois nécessiter une désactivation de ce fonctionnement. Pour cela, passez 1 dans le paramètre doubleClic.

A noter que seul le double-clic sera désactivé. Les sous-listes pourront toujours être déployées ou contractées par un clic sur l'icône de déploiement.

Si vous passez 0 ou omettez ce paramètre, le fonctionnement par défaut est appliqué.

COMPRESSER IMAGE (Thème Images)

COMPRESSER IMAGE (image; type; qualité)

A compter de la version 6.7, 4^e Dimension tire parti de QuickTime 4, sous MacOS et sous Windows. Par conséquent, la commande COMPRESSER IMAGE fonctionne désormais à l'identique sous MacOS et sous Windows.

L'utilisation de cette commande sous Windows nécessite la présence de la version 4 (ou supérieure) de QuickTime pour Windows sur le poste. Dans le cas contraire, elle sera sans effet. En outre, si aucune version de QuickTime n'est installée, l'erreur -9955 est générée.

- Notes*
- Les commandes utilisant QuickTime mais faisant appel à des fichiers disque (CHARGER ET COMPRESSER IMAGE et COMPRESSER FICHIER IMAGE) ne fonctionnent pas sous Windows.
 - 4D 6.7 propose également de nouvelles commandes de gestion des images, reportez-vous au [paragraphe "Images", page 48](#).

**CREER ENSEMBLE
SUR TABLEAU
(Thème Ensembles)**

CREER ENSEMBLE SUR TABLEAU (*table*; tabEnrg{; nomEns})

Paramètres	Type	Description
<i>table</i>	Table	→ Table de l'ensemble
tabEnrg	Tab Entier long Tab Booléen	→ Tableau de n° d'enregistrements, ou Tableau de booléens (Vrai = l'enregistre- ment est dans l'ensemble, Faux = il n'est pas dans l'ensemble)
nomEns	Alpha	→ Nom de l'ensemble à créer, ou Appli- quer la commande à l'ensemble User- set si ce paramètre est omis ou vide

Dans les versions précédente de 4D, le paramètre *table* était optionnel. Cette syntaxe introduisait une ambiguïté impossible à résoudre par 4D Compiler. Le paramètre *table* n'est désormais plus optionnel.

**CREER SELECTION
SUR TABLEAU
(Thème Sélections
temporaires)**

CREER SELECTION SUR TABLEAU (*table*; tabEnrg{; tempo})

Paramètres	Type	Description
<i>table</i>	Table	→ Table de la sélection
tabEnrg	Tab Entier long Tab Booléen	→ Tableau de n° d'enregistrements, ou Tableau de booléens (Vrai = l'enregistre- ment est dans la sélection, Faux = il n'y est pas)
tempo	Alpha	→ Nom de la sélection temporaire à créer, ou Appliquer la commande à la sélec- tion courante si ce paramètre est omis ou vide

Dans les versions précédentes de 4D, le paramètre *table* était optionnel. Cette syntaxe introduisait une ambiguïté impossible à résoudre par 4D Compiler. Le paramètre *table* n'est désormais plus optionnel.

DEPLACER OBJET
(Thème Propriétés
des objets)

DEPLACER OBJET({*; }objet; dépH; dépV{; redimH{; redimV{; *}}})

La commande DEPLACER OBJET permet de déplacer ou de redimensionner des objets du formulaire courant.

A compter de la version 6.7 de 4D, cette commande fonctionne dans le contexte des événements formulaire d'impression. Ce mode de fonctionnement permet de modifier dynamiquement l'apparence des formulaires avant leur impression. Il est désormais possible, en particulier, de construire des formulaires d'états dynamiques personnalisés.

**Dossier ACI (Thème
Environnement 4D)**

Suite au changement de raison sociale de la société ACI, la commande Dossier ACI est renommée Dossier 4D.

Dossier systeme
(Thème
Environnement
système)

Dossier systeme({type}) → Alpha

Paramètres	Type		Description
<i>type</i>	Entier long	→	Type de dossier système
Résultat	Alpha	←	Chemin d'accès à un dossier du système actif

Cette fonction admet un nouveau paramètre, optionnel, permettant de désigner le type de dossier système dont vous souhaitez obtenir le chemin d'accès.

Vous passez dans *type* un code représentant le type de dossier souhaité. 4D fournit les constantes prédéfinies suivantes (placées dans le thème "Documents système") :

Constante	Type	Valeur
Système	Entier long	0
Polices	Entier long	1
Préférences_Tous	Entier long	2
Préférences	Entier long	3
Ouverture au démarrage_Tous	Entier long	4
Ouverture au démarrage	Entier long	5
Ouverture à l'extinction_Tous	Entier long	6
Ouverture à l'extinction	Entier long	7
Menu Pomme ou Démarrer_Tous	Entier long	8
Menu Pomme ou Démarrer	Entier long	9

Constante	Type	Valeur
Extensions Mac	Entier long	10
Tableaux de bord Mac	Entier long	11

Les deux dernières constantes (Extensions Mac et Tableaux de bord Mac) sont réservées à une utilisation sous MacOS. Lorsqu'elles sont utilisées sous Windows, Dossier système retourne une chaîne vide.

L'emplacement de certains dossiers peut être différent suivant le type de session ouverte par l'utilisateur. Les constantes 2 à 9 permettent de définir si vous souhaitez obtenir le chemin d'accès du dossier spécifique à l'utilisateur courant (constantes simples) ou commun à tous les utilisateurs (constantes suivies de "Tous").

Si vous omettez le paramètre type, la fonction retourne l'emplacement du dossier Système actif (= constante Système).

Note Sous Windows, cette commande nécessite la présence du fichier système "SHFolder.DLL" version 4.71 minimum dans le dossier système actif de Windows. Sinon, la commande retourne des chaînes vides pour tous les types de dossiers, hormis pour le type 0 (Système).

FIXER ENTETE HTTP (Thème Serveur Web)

FIXER ENTETE HTTP (entete / *tabChamps* {; *tabValeurs*})

Paramètres	Type	Description
entete	Texte	→ En-tête HTTP de la requête
<i>tabChamps</i>	tableau Texte	→ Champs de l'en-tête HTTP
<i>tabValeurs</i>	tableau Texte	→ Contenu des champs de l'en-tête HTTP

La commande FIXER ENTETE HTTP permet d'écrire l'en-tête HTTP de la réponse du serveur Web 4D à la requête en cours de traitement. Cette commande admet désormais deux syntaxes.

Note Cette commande fonctionne uniquement en mode sans contexte.

- En version 6.5 de 4D, seule la syntaxe FIXER ENTETE HTTP(entete) était disponible. Cette syntaxe permettait d'écrire un en-tête du type "HTTP/1.0 200 OK"+Caractere(13)+"Set-Cookie: C=HELLO". Chaque champ d'en-tête est séparé par une séquence CR ou CR+LF (*Retour chariot+Retour à la ligne*), sous Windows et MacOS, 4D se charge du formatage de la réponse.

- A compter de la version 6.7 de 4D, une nouvelle syntaxe est disponible : l'en-tête HTTP peut être défini à l'aide de deux tableaux, tabChamps et tabValeurs.
L'en-tête sera écrit de la manière suivante :

tabChamps{1} = "X-VERSION" tabValeurs{1} = "HTTP/1.0" *
tabChamps{2} = "X-STATUS" tabValeurs{2} = "200 OK" *
tabChamps{3} = "Set-Cookie" tabValeurs{3} = "C=HELLO"

* Ces deux premiers éléments constituent la première ligne de la réponse. Lorsqu'ils sont saisis, ils doivent impérativement être les éléments 1 et 2 des tableaux. Il est toutefois possible de les omettre et d'écrire seulement — 4D se chargeant de formater l'en-tête :
tabChamps{1} = "Set-Cookie" tabValeurs{1} = "C=HELLO"

Conformément à la norme HTTP, les noms des champs HTTP sont toujours libellés en anglais.

Note Si plusieurs appels à FIXER ENTETE HTTP ont lieu dans le même process Web, seul le dernier appel est pris en compte.

Enfin, dans la version 6.7 de 4D, l'accès aux champs HTTP a été étendu : les champs Expires, Last-Modified et Content-Type sont désormais modifiables.
Seuls les champs Server, Date et Content-Length restent fixés par 4D.

FIXER INTERFACE
(Thème Interface
utilisateur)

- FIXER INTERFACE (interface)
- Les constantes du thème “Interfaces de plate-forme” ont été modifiées afin de refléter les évolutions des systèmes d'exploitation :
- La nouvelle constante Thème Mac a été ajoutée.
 - La constante Macintosh a été renommée en Mac Système 7.

Vous pouvez désormais passer dans le paramètre interface une des constantes suivantes :

Constante	Type	Valeur
Plate-forme automatique	Entier long	-1
Mac Système 7	Entier long	0
Windows NT 3.51	Entier long	1
Windows 9x	Entier long	2
Platinum	Entier long	3
Thème Mac	Entier long	4

La constante Thème Mac permet d'utiliser l'interface définie dans le *Gestionnaire d'apparence*. Ce gestionnaire n'existe que sous MacOS. Lorsqu'une base définie en Thème Mac est affichée sous Windows, l'interface "Windows 9x" est utilisée.

Note Cette valeur peut également être appliquée via les Propriétés de la base. Pour plus d'informations, reportez-vous au [paragraphe "Interface de plate-forme", page 18](#).

**FIXER PARAMETRE
BASE**
(Thème Définition
structure)

FIXER PARAMETRE BASE ({table;} sélecteur; valeur)

Cette commande admet de nouveaux sélecteurs, permettant de régler par programmation le gestionnaire interne des appels système de 4D, de modifier dynamiquement le timeout des connexions client/serveur, de faciliter paramétrage d'un serveur Web compilé et fusionné, ainsi que de définir le nombre total de process Web autorisés.

En outre, deux nouvelles valeurs sont disponibles pour le sélecteur 8, correspondant à la constante Mode conversion Web.

Nouveaux sélecteurs

Les nouveaux sélecteurs peuvent être définis à l'aide des constantes suivantes :

Constante	Type	Valeur
Appels système 4e Dimension	Entier long	10
Appels système 4D Server	Entier long	11
Appels système 4D Client	Entier long	12
Timeout 4D Server	Entier long	13
Timeout 4D Client	Entier long	14
Numéro du port	Entier long	15
Adresse IP d'écoute	Entier long	16
Jeu de caractères	Entier long	17
Process Web simultanés maxi	Entier long	18

- sélecteur = 10 (Appels système 4e Dimension) : les réglages s'appliquent à 4^e Dimension (monoposte) lorsque la commande est appelée depuis 4^e Dimension, ainsi qu'à 4D Tools.
- sélecteur = 11 (Appels système 4D Server) : les réglages s'appliquent à 4D Server lorsque la commande est appelée depuis 4D Server.
- sélecteur = 12 (Appels système 4D Client) : les réglages s'appliquent à 4D Client lorsque la commande est appelée depuis 4D Client.

Dans les trois cas, le paramètre valeur est de la forme hexadécimale 0x00aabbcc dans laquelle :

- aa = nombre minimum de ticks par appel au système (de 0 à 100 inclus)
- bb = nombre maximum de ticks par appel au système (de 0 à 100 inclus)
- cc = nombre de ticks entre deux appels au système (de 0 à 20 inclus).

Si une des valeurs est hors de son intervalle, 4D la fixe à son maximum.

Vous pouvez également passer dans valeur une des trois valeurs suivantes, correspondant à un ensemble de réglages standard :

- valeur = -1 : priorité maximum allouée à 4D,
- valeur = -2 : priorité moyenne allouée à 4D,
- valeur = -3 : priorité minimum allouée à 4D.

Note Les fréquences des appels peuvent également être réglées dans les Propriétés de la base (pour plus d'informations, reportez-vous au [paragraphe "Appel au système", page 22](#)).

- ▼ Cette méthode permet de définir les valeurs du minuteur si l'application courante est 4^e Dimension monoposte :

C_ENTIER LONG(\$ticksbtwcalls;\$maxticks;\$minticks;\$lparams)

Si (Type application=4e Dimension) ` Si nous sommes en 4D monoposte

\$ticksbtwcalls:=12

\$maxticks:=20

\$minticks:=7

\$lparams:=(\$minticks<<16)|(\$maxticks<<8)|\$ticksbtwcalls

FIXER PARAMETRE BASE(Appels système 4e Dimension;\$lparams)

Fin de si

- sélecteur = 13 (Timeout 4D Server) : permet de modifier la valeur du délai avant déconnexion (*timeout*) accordé par 4D Server aux postes clients.

Par défaut, la valeur du délai avant déconnexion utilisée par 4D Server est définie dans la page "Connexions" de la boîte de dialogue des Propriétés de la base, sur le poste serveur.

Le sélecteur Timeout 4D Server vous permet de fixer, à l'aide du paramètre valeur, un nouveau *timeout*, exprimé en minutes. Cette possibilité permet en particulier d'augmenter la valeur du *timeout* avant l'exécution sur le poste client d'une opération bloquante et de longue durée, risquant d'entraîner une déconnexion ; par exemple, l'impression d'un grand nombre de pages.

Vous disposez en outre de deux possibilités :

- effectuer une modification *globale* et *permanente* : la nouvelle valeur s'applique à tous les process et est stockée dans les préférences de l'application (équivalent à une modification de la valeur dans la boîte de dialogue des Propriétés de la base).
Pour cela, passez une valeur positive dans le paramètre valeur.
- effectuer une modification *restreinte* et *temporaire* : la nouvelle valeur ne s'applique qu'au process appelant (les autres process conservant la valeur d'origine), et est abandonnée dès que le serveur reçoit un signe d'activité du poste client — par exemple, dès que l'opération est terminée. Cette possibilité est utile pour gérer les opérations longues initiées par des plug-ins.
Pour cela, passez une valeur négative dans le paramètre valeur.

Pour reprendre l'exemple cité auparavant, l'appel de l'instruction suivante pourra éviter le problème :

```
`Augmentation du timeout à 3 heures pour le process courant
FIXER PARAMETRE BASE(Timeout 4D Server;-60*3)
`Exécution d'une opération longue hors du contrôle de 4D
...
WR MAILING (LaZone;3;0)
...
```

Note Pour définir une connexion “Ouvverte en permanence”, passez 0 dans valeur.

- sélecteur = 14 (Timeout 4D Client) : permet de modifier la valeur du délai avant déconnexion (*timeout*) accordé par le poste 4D Client au poste 4D Server.
Par défaut, la valeur du délai avant déconnexion utilisée par 4D Client est définie dans la page “Connexions” de la boîte de dialogue des Propriétés de la base, sur le poste client.
Pour plus d'informations sur le fonctionnement de ce sélecteur, reportez-vous à la description du sélecteur Timeout 4D Server (13).
Le sélecteur Timeout 4D Client est à utiliser dans des cas très spécifiques, car, de manière générale, il est déconseillé d'effectuer sur le poste des opérations bloquantes et de longue durée.
- sélecteur = 15 (Numéro du port) : permet de modifier par programmation le numéro du port TCP utilisé par le serveur Web 4D.
Par défaut, la valeur est 80.

Le numéro de port TCP est défini dans la page “Serveur Web I” de la boîte de dialogue des Propriétés de la base.

Le sélecteur Numéro du port est utile dans le cadre de serveurs Web 4D compilés et fusionnés avec 4D Engine (pas d'accès au mode Structure). Pour plus d'informations sur le numéro de port TCP, reportez-vous à la section “Services Web, Configuration” dans le manuel *Langage* de 4D.

- sélecteur = 16 (Adresse IP d'écoute) : permet d'indiquer par programmation l'adresse IP sur laquelle le serveur Web doit recevoir les requêtes HTTP. Par défaut, aucune adresse particulière n'est spécifiée (valeur = 0)

Ce paramètre est défini dans la page “Serveur Web I” de la boîte de dialogue des Propriétés de la base.

Le sélecteur Adresse IP d'écoute est utile dans le cadre de serveurs Web 4D compilés et fusionnés avec 4D Engine (pas d'accès au mode Structure).

Passez dans le paramètre valeur l'adresse IP sous forme hexadécimale. Ainsi, pour désigner une adresse du type “a.b.c.d”, le code sera de la forme :

```
C_ENTIER LONG($addr)
$addr:=( $a<<24)|($b<<16)|($c<<8)|$d
FIXER PARAMETRE BASE(Adresse IP d'écoute;$addr)
```

Par exemple, l'adresse 192.193.194.195 sera fixée avec l'instruction :
FIXER PARAMETRE BASE(Adresse IP d'écoute;0xC0C1C2C3)

Pour plus d'informations sur la désignation de l'adresse IP, reportez-vous à la section “Services Web, Paramétrages du Serveur Web” dans le manuel *Langage* de 4D.

- sélecteur = 17 (Jeu de caractères) : permet d'indiquer par programmation le jeu de caractères que le serveur Web 4D doit utiliser pour communiquer avec les navigateurs Web qui se connectent à la base. La valeur par défaut dépend de la langue du système d'exploitation.

Ce paramètre est défini dans la page “Serveur Web II” de la boîte de dialogue des Propriétés de la base. Le sélecteur Jeu de caractères est utile dans le cadre de serveurs Web 4D compilés et fusionnés avec 4D Engine (pas d'accès au mode Structure).

Les valeurs possibles sont les suivantes :

- 0 : Occidental
 - 1 : Japonais
 - 2 : Chinois
 - 3 : Coréen
 - 4 : Défini par l'utilisateur
 - 5 : *Réservé*
 - 6 : Europe Centrale
 - 7 : Cyrillique
 - 8 : Arabe
 - 9 : Grec
 - 10 : Hébreu
 - 11 : Turc
 - 12 : Nordique
- sélecteur = 18 (Process Web simultanés maxi) : permet de définir la limite strictement supérieure du nombre de process Web de tout type (contextuels, non contextuels ou appartenant à la réserve¹ de process) acceptés par le serveur Web. Lorsque ce nombre limite (moins un) est atteint, 4D ne crée plus de nouveau process et retourne le message "Serveur non disponible" (statut HTTP 503 - Service Unavailable) à toute nouvelle requête.
- Ce paramètre permet de prévenir la saturation du serveur Web 4D pouvant se produire lors d'un envoi massif de requêtes ou d'une demande excessive de création de contextes.
- La valeur par défaut est 32 000. Vous pouvez passer toute valeur incluse entre 10 et 32 000.

Note Si vous passez une valeur inférieure à la limite supérieure de la "réserve" de process (sélecteur 7), cette limite est réduite à la valeur du sélecteur 18. Si nécessaire, la limite inférieure de la réserve (sélecteur 6) est également ajustée.

Ce paramètre peut également être défini dans les Propriétés de la base. Pour plus d'informations sur ce point, notamment pour savoir comment fixer la valeur de ce paramètre, reportez-vous au [paragraphe "Nombre maximum de process Web", page 24](#).

1. La "réserve" de process est définie via les sélecteurs 6 et 7. Pour plus d'informations, reportez-vous au manuel *Langage* de 4D.

Nouvelles valeurs pour le sélecteur 8

Par défaut, le serveur Web 4D 6.7 utilise les feuilles de style en cascade (CSS1) pour générer des pages HTML dont l'apparence est très proche de celle des formulaires obtenus dans 4^e Dimension.

Dans le cas des bases créées avec des versions antérieures de 4D, ce fonctionnement peut perturber la conversion correcte des formulaires. Vous pouvez donc modifier le mode de conversion des formulaires.

Pour cela, le sélecteur 8 (Mode conversion Web) admet deux nouvelles valeurs (en *caractères italique* dans ce tableau) :

Mode	Description
1	Conversion au format HTML 2.0 (compatibilité version 6.0.x)
2	Conversion au format HTML 3.2 (compatibilité version 6.5.x)
3	<i>Conversion au format HTML 4.0 + CSS-P (depuis version 6.5.2)</i>
0	<i>Conversion au format HTML 4.0 si le browser le permet. Sinon, format HTML 3.2 + usage de tableaux.</i>

Par défaut, le mode 0 est utilisé dans 4D 6.7. Le paramétrage du mode de conversion Web s'effectue au niveau du process Web dans lequel la commande `FIXER PARAMETRE BASE` est appelée et est disponible en mode contextuel uniquement.

Note Pour plus d'informations sur la conversion des formulaires avec 4D 6.7, reportez-vous au [paragraphe "HTML 4.0 et CSS1 - Feuilles de style en cascade, niveau 1"](#), page 88.

LIRE IMAGE DANS BIBLIOTHEQUE (Thème Images)

LIRE IMAGE DANS BIBLIOTHEQUE (reflImage | *nomImage*; image)

Paramètres	Type	Description
reflImage <i>nomImage</i>	Num Alpha →	Numéro de référence ou Nom d'une image de la bibliothèque
image	Image ←	Image de la bibliothèque d'images

Cette commande admet désormais un nom d'image (alphanumérique) comme premier paramètre. Ce fonctionnement permet aux développeurs de composants de s'assurer que leurs appels d'images personnalisées en provenance de la bibliothèque d'images seront correctement effectués, même si les numéros de ces images ont été modifiés au moment de l'installation.

En effet, lorsqu'un composant utilisant des images est installé par 4D Insider, le programme peut renuméroter automatiquement les nouvelles images si des images de même numéro de référence existent déjà dans la base.

Note Pour plus d'informations sur les *composants* dans 4^e Dimension 6.7, reportez-vous au [paragraphe "Utilisation des composants 4D"](#), page 27, et à la documentation de 4D Insider.

Lire interface
(Thème Interface utilisateur)

Lire interface → Numérique

Les constantes du thème "Interfaces de plate-forme" pouvant être retournées par cette fonction ont été modifiées afin de refléter les évolutions des systèmes d'exploitation. Pour plus d'informations, reportez-vous à la description de la [routine FIXER INTERFACE](#) (Thème Interface utilisateur), page 74.

Lire parametre base
(Thème Définition structure)

Lire parametre base ({table;} sélecteur) → Entier long

Cette commande admet plusieurs nouveaux sélecteurs, permettant de lire les réglages courants du gestionnaire interne des appels système de 4D, les valeurs de timeout client/serveur, et divers paramètres du serveur Web de 4D.

Les nouveaux sélecteurs peuvent être définis à l'aide des constantes suivantes :

Constante	Type	Valeur
Appels système 4e Dimension	Entier long	10
Appels système 4D Server	Entier long	11
Appels système 4D Client	Entier long	12
Timeout 4D Server	Entier long	13
Timeout 4D Client	Entier long	14
Numéro du port	Entier long	15
Adresse IP d'écoute	Entier long	16
Jeu de caractères	Entier long	17
Process Web simultanés maxi	Entier long	18

Pour plus d'informations sur ces paramètres, reportez-vous à la description de la [routine FIXER PARAMETRE BASE](#) (Thème Définition structure), page 75.

- ▼ Cette méthode permet de récupérer les valeurs courantes du minuteur interne de 4D :

```
C_ENTIER LONG($ticksbtwcalls;$maxticks;$minticks;$lparams)
Si (Type application=4e Dimension) ` Si nous sommes en 4D monoposte
  $lparams:=Lire parametre base(Appels système 4e Dimension)
  $ticksbtwcalls:=$lparams & 0x00ff
  $maxticks:=( $lparams>>8) & 0x00ff
  $minticks:=( $lparams>>16) & 0x00ff
Fin de si
```

- ▼ Le sélecteur 16 (Adresse IP d'écoute) permet d'obtenir l'adresse IP sur laquelle le serveur Web 4D reçoit les requêtes HTTP. L'adresse obtenue est de forme hexadécimale. L'exemple suivant permet de décomposer l'adresse IP reçue :

```
C_ENTIER LONG($a;$b;$c;$d)
C_ENTIER LONG($addr)
$addr:=Lire parametre base(Adresse IP d'écoute)
$a:=( $addr>>24)&0x000000ff
$b:=( $addr>>16)&0x000000ff
$c:=( $addr>>8)&0x000000ff
$d:=$addr&0x000000ff
```

**LIRE PROPRIETES
CHAMP**
(Thème Définition
structure)

LIRE PROPRIETES CHAMP (chpPtr | tableNum{; champNum};
champType{; champLong{; indexé{; unique{; invisible}}})

Paramètres	Type	Description
chpPtr tableNum	Pointeur Num	→ Pointeur de champ ou Numéro de table
champNum	Numérique	→ Numéro de champ si un numéro de table est passé en premier paramètre
champType	Numérique	← Type du champ
champLong	Numérique	← Longueur du champ (si alpha)
indexé	Booléen	← Vrai = Indexé, Faux = Non indexé
unique	Booléen	← Vrai = Unique, Faux = Non unique
invisible	Booléen	← Vrai = Invisible, Faux = Visible

Note Pour des raisons de cohérence avec les autres commandes du thème “Définition structure”, la commande PROPRIETES CHAMP a été renommée en LIRE PROPRIETES CHAMP.

Cette commande retourne désormais deux informations supplémentaires sur le champ désigné par tableNum et champNum ou par chpPtr :

- Le paramètre unique retourne Vrai si le champ dispose de l'attribut "Unique", Faux sinon. L'attribut Unique ne peut être appliqué qu'aux champs indexés.
- Le paramètre invisible retourne Vrai si le champ dispose de l'attribut "Invisible", Faux sinon. L'attribut Invisible permet de masquer le champ dans les éditeurs standard de 4D (étiquettes, graphes...).

**LIRE PROPRIETES
LISTE**
(Thème Listes
hiérarchiques)

LIRE PROPRIETES LISTE(liste; apparence{; icône{; ligneHauteur
{;doubleClic}}})

Paramètres	Type		Description
liste	RefListe	→	Numéro de référence de la liste
apparence	Numérique	←	Style graphique de la liste
icône	Numérique	←	Référence de ressource MacOS 'cicn' ou 0 = icône par défaut de la plate-forme
ligneHauteur	Numérique	←	Hauteur minimale de la ligne (pixels)
doubleClic	Entier long	←	Déploiement/contraction sur double-clic 0 = autorisé, 1= empêché

La commande LIRE PROPRIETES LISTE retourne, dans le nouveau paramètre optionnel doubleClic, la valeur de cette option.

Si doubleClic vaut 1, le déploiement ou la contraction des sous-listes en cas de double-clic sur l'élément parent est désactivé. Si doubleClic vaut 0, ce fonctionnement est actif (valeur par défaut).

Pour plus d'informations sur ce paramètre, reportez-vous à la [routine CHANGER PROPRIETES LISTE \(Thème Listes hiérarchiques\)](#), page 70.

OUVRIR URL WEB
(Thème Serveur
Web)

OUVRIR URL WEB (url {; *})

Sous MacOS, cette commande tient désormais compte des réglages Internet (navigateur par défaut) définis dans le Gestionnaire de configuration ou *Configuration Manager*, lorsque ceux-ci existent.

PICT VERS GIF
(Thème Images)

Pour des raisons de cohérence avec les autres commandes du thème "Images", la commande PICT VERS GIF a été renommée en IMAGE VERS GIF.

SIECLE PAR DEFAULT
(Thème Dates et heures)

SIECLE PAR DEFAULT (siècle{; anPivot})

Lorsque cette commande n'est pas spécifiquement appelée, le fonctionnement par défaut de 4^e Dimension a été modifié. Dans les versions précédentes, le programme présumait que les dates appartenaient au 20^e siècle :

- pour la saisie 25/01/97, 4D interprétait "25 janvier 1997".
- pour la saisie 25/01/07, 4D interprétait "25 janvier 1907".

Désormais en version 6.7, 4D admet par défaut la valeur 30 comme année pivot (ce qui équivaut à exécuter l'instruction SIECLE PAR DEFAULT (19;30)).

Ainsi, par défaut avec 4^e Dimension version 6.7 :

- pour la saisie 25/01/97, 4D interprète "25 janvier 1997".
- pour la saisie 25/01/30, 4D interprète "25 janvier 1930".
- pour la saisie 25/01/29, 4D interprète "25 janvier 2029".
- pour la saisie 25/01/07, 4D interprète "25 janvier 2007".

**SUPPRIMER IMAGE
DANS
BIBLIOTHEQUE**
(Thème Images)

SUPPRIMER IMAGE DANS BIBLIOTHEQUE (refImage | *nomImage*)

Paramètres	Type	Description
refImage <i>nomImage</i>	Num Alpha →	Numéro de référence ou Nom d'une image de la bibliothèque

Cette commande admet désormais un nom d'image (alphanumérique) comme paramètre.

Pour plus d'informations, reportez-vous au [paragraphe "LIRE IMAGE DANS BIBLIOTHEQUE \(Thème Images\)"](#), page 80.

TABLEAU ENTIER LONG SUR SELECTION (Thème Tableaux)

TABLEAU ENTIER LONG SUR SELECTION (*table*; tabEnrg{; tempo))

Paramètres	Type	Description
<i>table</i>	Table	→ Table de la sélection courante
tabEnrg	Tab Entier long	→ Tableau de n° d'enregistrements
tempo	Alpha	→ Nom de la sélection temporaire ou Sélection courante si ce paramètre est omis

Dans les versions précédente de 4D, le paramètre *table* était optionnel. Cette syntaxe introduisait une ambiguïté impossible à résoudre par 4D Compiler. Le paramètre *table* n'est désormais plus optionnel.

Types de fenêtres (Thème Fenêtres)

La constante Fenêtre palette (thème "Créer fenetre") a été modifiée : sa valeur est désormais 1984. Cette nouvelle valeur permet de créer sous MacOS une fenêtre de type palette ayant les mêmes caractéristiques que la précédente, tout en respectant les attributs d'apparence définis par l'intermédiaire du *Gestionnaire d'apparence*. La valeur précédente de la constante (720) crée une palette ne tenant pas compte de l'apparence MacOS courante.

Sous Windows, cette modification n'a pas d'influence sur l'apparence ni le fonctionnement des fenêtres de ce type.

Note Les valeurs des constantes des types de fenêtres utilisées par 4D sont issues des spécifications d'Apple. Pour obtenir la liste complète des types disponibles sous MacOS, reportez-vous à la documentation Apple.

4

Serveur Web

La version 6.7 de 4^e Dimension propose de nombreuses nouveautés liées au serveur Web de 4D. Ces nouveautés s'articulent autour des points suivants :

- **Compatibilité avec les technologies Internet** : le serveur Web 4D permet désormais d'exploiter les technologies Internet avancées, telles que le XML, les feuilles de style en cascade, en encore les CGI.
- **Interface avec d'autres serveurs HTTP** : le serveur Web 4D peut désormais s'interfacer avec de nombreux autres serveurs HTTP, notamment via la technologie ISAPI.
- **Gestion des pages HTML** : la gestion des balises HTML (tags) insérées dans les pages Web 4D a été améliorée, et de nouvelles balises sont disponibles.
- **Support des connexions Internet sécurisées** : il est désormais possible de sécuriser les communications réseau 4D via le protocole SSL.

Note Plusieurs commandes ont été ajoutées ou modifiées dans le thème "Serveur Web" du langage de 4D 6.7 (cf. page 31). Le nouveau thème "Protocole sécurisé" (cf. page 57) comprend les commandes chargées de la gestion du protocole SSL au sein de 4D.

En outre, diverses nouveautés contribuent enrichir et à renforcer le serveur Web de 4D :

- De nouveaux paramètres Web et informations sont disponibles dans les Propriétés de la base (cf. [paragraphe "Propriétés de la base", page 22](#)) et via la [routine FIXER PARAMETRE BASE \(Thème Définition structure\), page 75](#).
 - Un nouvel assistant intégré de publication des données sur le Web (reportez-vous au manuel "4D Web Assistant"),
 - Enfin, les commandes Internet de 4D proposent plusieurs nouveautés (reportez-vous au manuel "4D Internet Commands").
-

Compatibilité avec les technologies Internet

WML

Le serveur Web 4D supporte désormais la technologie WML (*Wireless Markup Language*). Cette fonctionnalité autorise la consultation et la saisie d'informations dans des bases 4D à l'aide d'un téléphone mobile ou d'un assistant électronique personnel (PDA).

Note Le langage WML, associé au protocole WAP (*Wireless Application Protocol*), est développé conjointement par plusieurs sociétés. Le WAP propose un ensemble d'outils de communication réseau, destiné à permettre aux téléphones mobiles et aux PDA de visualiser du texte publié dans des pages Web. La technologie WML est ouverte et libre de droits. Pour plus d'informations sur les spécifications du WML, veuillez consulter le site Web de Phone.com : <http://www.phone.com/>

La visualisation et la saisie d'informations s'effectuent par l'intermédiaire de pages WML utilisant le mécanisme des balises 4DVAR ou 4DSCRIPT.

Voici la liste des documents WML acceptés par le serveur Web 4D :

Extension	Type MIME	Description
.wml	text/vnd.wap.wml	Pages WML (toujours analysées par 4D*)
.wmls	text/vnd.wap.wmlscript	Scripts WML (côté client)
.wmlc	application/vnd.wap.wmlc	Pages WML binaires
.wmlsc	application/vnd.wap.wmlscript	Scripts WML binaires
.wbmp	image/vnd.wap.wbmp	Images bitmap destinées aux mobiles (parfois non supportées)

* Permet l'insertion dynamique de données via les balises 4DVAR et 4DSCRIPT.

HTML 4.0 et CSS1 - Feuilles de style en cascade, niveau 1

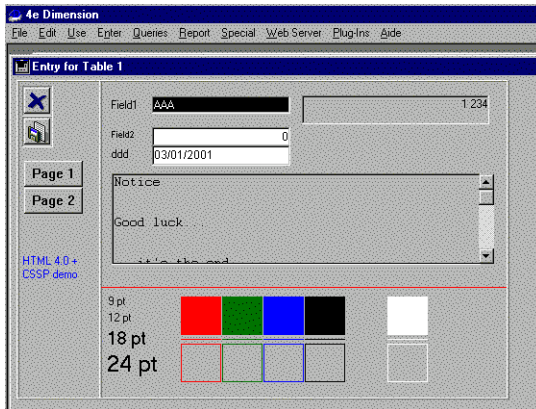
Les spécifications des CSS1 (*Cascading Style Sheet 1*) ont été définies par le consortium W3C (*World Wide Web Consortium*). Ces feuilles de style contiennent un ensemble de caractéristiques définissant l'apparence d'un document : police de caractères, taille, couleur, titres, texte courant, interlignage, etc.

Le serveur Web 4D utilise désormais les CSS1 pour générer des pages HTML dont l'apparence est très proche de celle des formulaires obtenus dans 4^e Dimension. 4D peut désormais afficher :

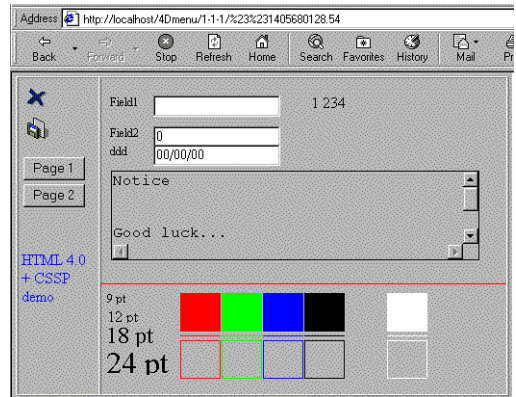
- des rectangles en conservant leur position, leur couleur¹ et leur largeur de ligne,
- la taille réelle de la police des zones de texte,
- les barres de défilement des zones de texte éditables.

Les écrans suivants illustrent les possibilités de conversion des formulaires 4D à l'aide des CSS :

Formulaire 4D



Browser Web



Pour des raisons de compatibilité, vous pouvez choisir, à l'aide de la commande `FIXER PARAMETRE BASE`, le mode de conversion Web à utiliser pour les formulaires. Pour cela, utilisez la syntaxe suivante :

`FIXER PARAMETRE BASE([table];Mode conversion Web;mode)`

1. Lorsque vous définissez un rectangle avec une couleur de bordure et de fond différentes, c'est la couleur de la bordure qui sera utilisée pour la totalité du rectangle dans le navigateur.

Le paramètre *mode* peut prendre l'une des valeurs suivantes :

Mode	Description
1	Conversion au format HTML 2.0 (compatibilité version 6.0.x)
2	Conversion au format HTML 3.2 (compatibilité version 6.5.x)
3	Conversion au format HTML 4.0 + CSS-P (depuis version 6.5.2)
0	Conversion au format HTML 4.0 si le browser le permet. Sinon, format HTML 3.2 + usage de tableaux.

Note Pour plus d'informations, reportez-vous au [paragraphe “FIXER PARAMETRE BASE \(Thème Définition structure\)”](#), page 75.

Lorsque l'algorithme de conversion 6.5 est utilisé (en HTML 4) :

- une police non proportionnelle est affichée dans le browser lorsque c'est le cas dans 4D.
- la largeur des zones de texte saisissables est mieux gérée.

Les CSS ne sont pas utilisées pour les objets pour lesquels elles ne sont pas nécessaires (tels que les listes hiérarchiques et les barres de menus). En outre, certains objets de formulaires comme les boutons et les zones de texte saisissables ne peuvent pas être contrôlés.

Depuis la version 6.5 de 4D, les documents .CSS sont envoyés avec le type MIME “text/css”. En mode contextuel et en mode sans contexte, ces documents ne sont pas analysés par 4D.

XML

Le serveur Web 4D accepte désormais les documents .xml, .xsl et .dtd. Ces documents sont respectivement envoyés avec le type MIME “text/xml”, “text/xsl” et “text/xml”.

Quel que soit le mode (contextuel ou sans contexte) depuis lequel ces documents sont envoyés, 4D analyse leur contenu et traite les éventuelles balises “4DVAR” ou “4DSCRIPT” qu'ils contiennent, afin de générer du XML dynamique.

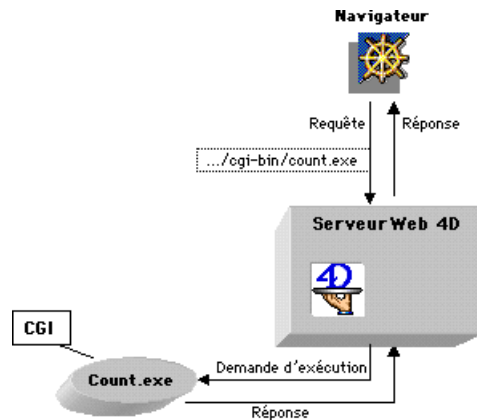
Note Il n'est pas possible d'envoyer du XML depuis l'intérieur même d'un formulaire 4D, en utilisant une balise du type {mapage.xml} insérée dans un texte statique.

Compatibilité avec les CGI

Le serveur Web 4D version 6.7 peut désormais utiliser des CGI (*Common Gateway Interface*)¹. Un CGI se présente sous la forme d'un programme exécutable, d'un script PERL ou d'une DLL s'interfaçant avec un serveur Web.

Les CGI sont aux pages Web ce que les plug-ins sont aux méthodes 4D. Appelé par le serveur Web, le CGI exécute une tâche et retourne une réponse — une page Web complète ou du code HTML venant s'insérer dans une page envoyée par le serveur. Des CGI sont fréquemment utilisés pour afficher les compteurs d'accès, gérer les livres d'or (*guestbook*), recevoir le résultat d'un formulaire (*form to mail*), etc.

Principe de fonctionnement d'un CGI



Une multitude de CGI sont aujourd'hui disponibles. La plupart d'entre eux sont dans le domaine public.

Installation des CGI sur le serveur Web 4D

L'appel d'un CGI s'effectue par l'intermédiaire d'un URL, d'une action ou d'un marqueur HTML inséré dans une page, en fonction de la tâche effectuée par le CGI. Dans tous les cas, la chaîne HTML doit contenir /cgi-bin/ suivi du nom du CGI et éventuellement d'un chemin d'accès (utilisant la syntaxe HTML) ainsi que d'une chaîne d'interrogation.

Par exemple, l'URL "http://195.1.2.3/cgi-bin/search.exe" provoquera l'exécution du CGI search.exe. De même, si vous placez le marqueur `` au sein d'une page HTML, le CGI counter.exe sera exécuté lors de l'envoi de la page.

1. Le terme CGI désigne, à l'origine, la norme définissant l'interfaçage des applications externes avec les serveurs HTTP. Par extension, CGI est aujourd'hui utilisé pour désigner ces applications externes elles-mêmes.

Pour pouvoir être appelés, les CGI doivent obligatoirement être placés à la racine d'un dossier nommé **cgi-bin**. Ce dossier doit lui-même être situé à la racine du serveur Web ou dans un sous-dossier. Il peut y avoir plusieurs dossiers **cgi-bin** par serveur. Ce dossier peut contenir d'autres fichiers que des exécutables, mais seuls ces derniers peuvent être appelés depuis un client Web.

- ▼ Voici des exemples d'emplacements et les URL pouvant être appelés :

Emplacement des éléments (racine du serveur Web)	URLs correspondants
Dossier [mabase]	
mabase.4db (structure)	(http://195.1.2.3/)
Dossier [cgi-bin]	
compteur.exe	(http://195.1.2.3/cgi-bin/compteur.exe)
Dossier [Divers]	
Dossier [cgi-bin]	
script.pl	(http://195.1.2.3/Divers/cgi-bin/script.pl)

Types de CGI

Les types de CGI utilisables sont différents sous Windows et sous MacOS.

- Sous Windows, les CGI peuvent être du type suivant :
 - des exécutables (.EXE) utilisant la “console” et les variables d'environnement. Le code source est généralement multi-plate-forme (Windows et Unix). Leurs noms sont de la forme *nnnn.exe* ou *nph-nnn.exe*.
Pour plus d'informations sur ce type de CGI, veuillez consulter le site <http://hoohoo.ncsa.uiuc.edu/cgi/>.
 - des DLL ISAPI, c'est-à-dire des extensions pour IIS (*Internet Information Server*)¹. Leurs noms sont de la forme *nnnn.dll* ou *nph-nnnn.dll*. Pour des raisons de performances, les DLL appelées de la sorte ne sont déchargées qu'à l'arrêt du serveur Web.
Pour plus d'informations sur ce type de CGI, veuillez consulter le site <http://www.microsoft.com/iis/>.

1. Voir aussi à ce sujet le [paragraphe “4DISAPI.DLL et NPH-CGI4D.EXE”](#), page 96.

- de scripts PERL utilisant la “console” et les variables d’environnement. Ces CGI nécessitent la présence d’un interpréteur permettant de les exécuter. Ils présentent toutefois l’avantage d’être entièrement multi-plate-forme (Windows, Unix et MacOS). Leurs noms sont de la forme *nnnn.pl*, *nph-nnnn.pl*, *nnnn.cgi* ou encore *nph-nnnn.cgi*.
Pour plus d’informations sur ce type de CGI, veuillez consulter le site <http://www.perl.com/>.
- Sous MacOS, les CGI peuvent être du type suivant :
 - des applications, extensions de 4D WebStar et MacHTTP. Leurs noms sont de la forme *nnnn.cgi* et *nnnn.acgi* (ce sont obligatoirement des applications).
Pour plus d’informations sur ce type de CGI, veuillez consulter le site <http://dev.starnine.com/>.
 - des scripts PERL. Ces CGI nécessitent la présence de MacPERL. Leurs noms sont de la forme *nnnn.pl*, *nph-nnnn.pl*, *nnnn.cgi* ou encore *nph-nnnn.cgi* (ce sont obligatoirement des fichiers texte).
Pour plus d’informations sur ce type de CGI, veuillez consulter le site <http://www.macperl.com/>.

Interaction entre le serveur Web 4D et les CGI

L’appel d’un CGI ne modifie jamais l’environnement de 4D (sélections, variables...).

4D n’impose aucune limite de taille de la réponse. Cependant, la durée maximum de traitement allouée au CGI est fixée à 30 secondes. Au-delà de ce délai, le serveur Web retournera une erreur.

Un CGI est toujours exécuté hors contexte, quel que soit le mode depuis lequel il a été appelé.

A noter toutefois qu’en mode contextuel, il est conseillé de ne pas utiliser de CGI qui renvoie du code HTML, car il y a dans ce cas risque de désynchronisation du contexte.

- Erreurs renvoyées par 4D lors d’un appel CGI (Windows et MacOS)
Lorsque l’appel d’un CGI génère une erreur, 4D retourne une des réponses suivantes, sous forme de page HTML standard :
 - *Non trouvé* : le CGI n’a pu être localisé par 4D, ou bien l’interpréteur PERL n’est pas installé
 - *Interdit* : le client Web demande autre chose qu’un exécutable dans un dossier [cgi-bin]
 - *Timeout* : le CGI n’a pu traiter la requête en 30 secondes

- *Mauvaise réponse* : la réponse du CGI n'a pu être traitée par 4D ou la DLL ISAPI a causé une exception
- *Erreur interne* : mémoire saturée, etc.
- **Protection de 4D vis-à-vis d'une DLL ISAPI (Windows uniquement)**
Bien qu'appelée directement, si une DLL ISAPI génère une exception (non définie par l'utilisateur), celle-ci n'entraîne pas l'arrêt de 4D. 4D renvoie la réponse "Mauvaise réponse" (la DLL a été incapable de répondre).

Informations à destination des développeurs de CGI

Ce paragraphe est principalement destiné aux programmeurs souhaitant développer des CGI pour leur bases 4D.

- **Variables d'environnement**
4D définit les variables d'environnement en conformité avec les spécifications CGI/1.1, avec les précisions suivantes :
 - GATEWAY_INTERFACE : toujours "CGI/1.1"
 - SERVER_SOFTWARE : toujours de la forme "4D_WebStar_D/version"
 - SERVER_PROTOCOL : toujours "HTTP/1.0"
 - SERVER_PORT_SECURE : contient "1" si la connexion HTTP est sécurisée, sinon "0".
 - PATH_TRANSLATED : contient le chemin d'accès complet de la racine HTML du serveur, auquel est ajouté la portion de chemin qui suit le nom du CGI. Par sécurité, la portion de chemin ne peut contenir les séquences // ou ..
Exemple : Racine du serveur C:/web
Pour un appel CGI du type /cgi-bin/cgi.exe/path, PATH_TRANSLATED vaut "C:/web/path". Pour un appel CGI du type /cgi-bin/cgi.exe/./path, 4D renvoie l'erreur *Interdit*.
 - REMOTE_IDENT : nom d'utilisateur, sinon non définie
 - HTTP_AUTHORIZATION, HTTP_CONTENT_LENGTH et HTTP_CONTENT_TYPE : non définies
 - ALL_HTTP et URL sont définies dans le cas d'appels de DLL ISAPI.
 - CERT_xxx et HTTPS_xxx sont définies si la connexion est sécurisée (pour les DLL uniquement).

En plus des variables d'environnement standard, 4D ajoute des variables texte du type FORMVAR_nomvariable :

- si la requête est envoyée avec la méthode "POST", ces variables correspondent aux zones de saisie du formulaire (par exemple FORMVAR_NOM, FORMVAR_PRENOM...) à l'exception des champs binaires (INPUT TYPE="FILE"). Ce système peut être utilisé avec les formulaires encodés "www/url-encoded" et "multipart/form-data".
- si la requête est envoyée avec la méthode "GET", ces variables correspondent aux valeurs passées par la chaîne d'interrogation (par exemple, dans le cas de l'URL .../cgi.exe?nom=martin&code=75, FORMVAR_NOM vaudra "martin" et FORMVAR_CODE vaudra "75").

Ce fonctionnement présente l'avantage de faciliter le traitement des formulaires (il n'est pas nécessaire d'analyser les chaînes a=1&b=2&...), mais rend le CGI spécifique à 4D.

■ Traitement des réponses retournées par les CGI

- Si le nom du CGI (exécutable Windows ou script PERL) débute par nph- (*No Parsing Header*), 4D retourne la réponse "telle quelle" au client Web. Dans ce cas, il revient au CGI de respecter la norme HTTP.

En ce qui concerne les DLL ISAPI, 4D n'analyse jamais la réponse, que le préfixe nph- soit présent ou non.

- Si ce n'est pas le cas, 4D se charge de renvoyer l'en-tête HTTP :
 - si "Content-Type" n'est pas spécifié par le CGI, 4D renvoie systématiquement "Content-Type: text/html",
 - si "Location" est spécifié, 4D ignore les autres éléments de la réponse et effectue une redirection HTTP,
 - si "Status" n'est pas spécifié, 4D renvoie "HTTP/1.0 200 OK".
 4D accepte tout type de changement de ligne (Windows-CRLF, MacOS-CR, Unix-LF) dans l'en-tête de la réponse HTTP et se charge de la reformater.
- dans le cas des DLL ISAPI, 4D accepte les traitements asynchrones (HttpExtensionProc retourne HSE_STATUS_PENDING). L'appel à ServerSupportFunction (HSE_REQ_DONE_WITH_SESSION) doit avoir lieu dans les 30 secondes. Si la fonction TerminateExtension est définie, elle est toujours appelée avec la valeur HSE_TERM_MUST_UNLOAD.
- dans le cas des CGI WebStar, 4D autorise l'envoi fractionné des réponses supérieures à 32 Ko.

Interface avec d'autres serveurs HTTP

4DISAPI.DLL et NPH-CGI4D.EXE

4^e Dimension version 6.7 est accompagné de deux nouvelles extensions, 4DISAPI.DLL et NPH-CGI4D. Le but de ces extensions est de permettre à un serveur HTTP de transmettre des requêtes au serveur Web de 4D. Avec ces extensions, 4D peut être interrogé par tout autre serveur HTTP. Ce mécanisme autorise en particulier le développement de systèmes dans lesquels un serveur Web 4D non sécurisé peut être interrogé via un autre serveur HTTP, tournant lui en mode sécurisé.

Note Ces deux extensions sont disponibles sous Windows uniquement.

L'extension 4DISAPI respecte les spécifications définies par ISAPI (Internet Services Application Programming Interface). La technologie ISAPI a été développée à l'origine par Microsoft® pour le serveur IIS, mais a été depuis rendue compatible avec de nombreux serveurs HTTP tels que Netscape®, Apache® ou Sambar®.

L'extension NPH-CGI4D.EXE respecte les spécifications des CGI (*Common Gateway Interface*) et peut être utilisée avec l'ensemble des serveurs compatibles CGI.

Le fonctionnement de ces deux extensions est identique. La compatibilité CGI est plus largement répandue parmi les serveurs HTTP, toutefois les performances des extensions CGI sont généralement inférieures à celles des extensions ISAPI.

Voici une liste indicative des serveurs HTTP compatibles avec les extensions ISAPI :

Editeur/Produit	Compatibilité ISAPI
Microsoft Corp/Internet Information Server	Oui (1.0 & 2.0)
Process Software/Software Purveyor	Oui (1.0)
Apache 1.3b	
Spyglass Software Server SDK 2.0	Oui
Cyber Presence/Secure SSL Internet Server	Oui
O'Reilly/WebSite 1.1g	Oui (2.0)
Computer Software Manufaktur GmbH/Alibaba	Oui
The Internet Factory/Commerce Builder	Oui
Luckman Interactive/Web Commander	Oui
Server Seven (beta v.51)	Oui
Zeus Technology (Unix ISAPI)	Oui
Sambar Server	Oui

Principe de fonctionnement

Le principe de fonctionnement de ces extensions est le suivant : un serveur HTTP "A" publie des pages sur Internet, et un autre serveur HTTP, "B", est par exemple un 4D Server utilisé en Intranet. Afin que les deux serveurs puissent communiquer, il vous suffit d'ajouter une extension 4DISAPI ou NPH-4DCGI dans le répertoire [Scripts] du serveur A.

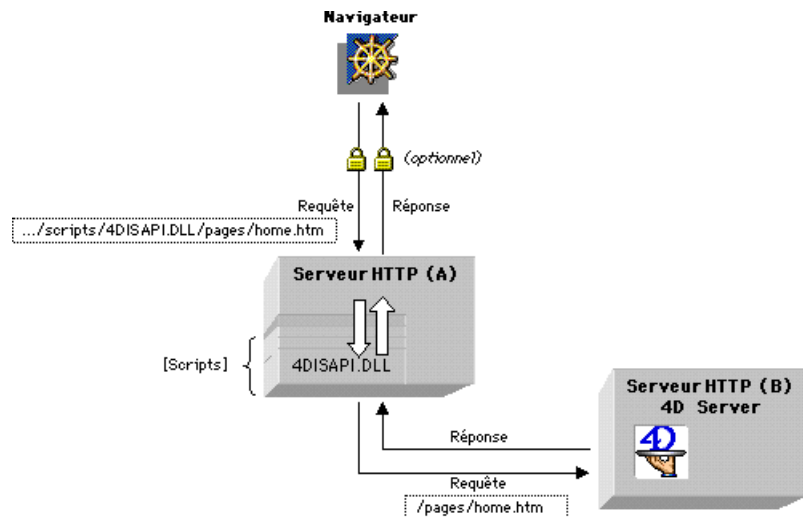
Lorsqu'un navigateur Web envoie une requête au serveur A, celui-ci la retransmet¹ au serveur B via l'extension 4D ISAPI ou NPH-4DCGI, par l'intermédiaire de l'URL. La réponse² est ensuite acheminée au navigateur en sens inverse. Le corps de la requête ou de la réponse HTTP n'est jamais modifié par les extensions.

La requête initiale envoyée au serveur A peut être effectuée en clair ou en mode sécurisé (SSL). La communication entre les deux serveurs HTTP et l'extension 4DISAPI.DLL s'effectue en clair.

Note Les extensions 4DISAPI et NPH-4DCGI ne sont pas compatibles avec le mode contextuel du serveur Web 4D.

Le schéma suivant illustre ce principe :

Exemple de fonctionnement de 4DISAPI.DLL



1. Généralement, l'extension transmet au serveur B la partie de l'URL située après son propre appel, y compris les éléments d'interrogation.
2. Lorsque le nom du CGI débute par NPH- (*No Parsing Header*), il n'est pas nécessaire que l'en-tête HTTP de la réponse soit analysé par le serveur, le CGI se chargeant lui-même du formatage.

- Les extensions reconnaissent les méthodes GET, HEAD et POST, elles gèrent les différents statuts renvoyés par 4D (*200 OK, 302 Moved Temporarily, 404 Not Found...*).
- Il n'est pas possible de procéder à une authentification au niveau HTTP via l'extension 4DISAPI ou NPH-CGI4D. Pour cela, il est nécessaire d'utiliser un formulaire HTML (ce qui est parfaitement envisageable en connexion sécurisée).

Note Les extensions sont avant tout destinées à être utilisées pour recevoir et renvoyer des données dynamiques, et en particulier pour poster des données. Le simple service de pages Web n'offre pas des performances optimales en cas de transit via des extensions ISAPI ou CGI.

Installation et configuration

L'installation des extensions 4DISAPI et NPH-CGI4D s'effectue par simple copie des fichiers 4DISAPI.DLL ou NPH-CGI4D.EXE dans le dossier [Scripts] du serveur HTTP.

Chaque extension installée s'accompagne d'un fichier de configuration (de type *.INI). Le fichier .INI doit porter le même nom que l'extension (par exemple 4DISAPI.INI). L'extension et son fichier de configuration doivent être placés dans le même dossier.

Vous pouvez configurer un serveur HTTP de manière à ce qu'il puisse cibler plusieurs autres serveurs HTTP. Dans ce cas, placez dans le dossier [Scripts] du serveur HTTP autant d'extensions que de serveurs-cibles. Il vous suffit de les renommer (par exemple 4DISAPI2.DLL, 4DISAPI3.DLL, etc.). Veillez à insérer un fichier de configuration par extension, et à le renommer en conséquence (4DISAPI2.INI, 4DISAPI3.INI, etc.).

Le fichier .INI est constitué d'une seule section : [Forward]. Cette section accepte les commandes suivantes :

- TargetServer =
Nom ou adresse IP du serveur Web à cibler (par exemple monserveur.net ou 192.193.194.195)¹. Laisser la chaîne vide pour repérer le serveur par son adresse si la résolution par nom n'est pas disponible. Le nom "localhost" est reconnu comme étant l'adresse 127.0.0.1. Par défaut, l'adresse 127.0.0.1 est utilisée.

1. Dans la version 6.7 de 4D, seul l'adressage IPv4 (xxx.yyy.zzz.ttt) est possible. Le support de l'adressage IPv6 (a:b:c:d:e:f:g:h:i) est en cours de réalisation.

- **TargetPort =**
Port sur lequel le serveur-cible écoute (par exemple 81). Par défaut, le port 8080 est utilisé.
- **Timeout =**
Délai maximum d'attente de la réponse du serveur (exprimé en secondes). La valeur par défaut est de 30 secondes.
- **Allowed =**
Liste des URL autorisés, séparés par des virgules. Par exemple : /pages, /img pour n'autoriser que les URL commençant par /pages et /img. Pour autoriser la totalité du site, inscrivez une barre oblique seule / (paramétrage par défaut).
- **Forbidden =**
Liste des URL interdits, séparés par des virgules. Par exemple : /4dmethod, /pages2 pour interdire les URL commençant par /4dmethod et /pages2. Pour ne rien interdire, ne saisissez rien dans la liste (paramétrage par défaut).

Compte tenu des exemples d'autorisations et d'exclusions fournis ci-dessus, les URL suivants du serveur-cible seront accessibles ou non :

/pages/document.html	accessible
/pages1/onepage.html	accessible
/www/image.gif	inaccessible
/pages2/mypage.html	inaccessible
/4dmethod/myproc	inaccessible

Si un URL déclaré interdit est sollicité, l'extension retourne directement l'erreur "*HTTP/1.0 403 Forbidden*".

Utilisation

Les extensions 4DISAPI et NPH-CGI4D acceptent les URLs suivants :

- **Appel de 4D** (4D ne reçoit que la portion de chemin qui suit le nom de l'extension) :
http://adresse-serveur/cgi-bin/4disapi.dll/[chemin d'accès]
http://adresse-serveur/cgi-bin/nph-cgi4d.exe/[chemin d'accès]
- **Test de fonctionnement de l'extension** (écho de la requête) :
http://adresse-serveur/cgi-bin/4disapi.dll/~-echo
http://adresse-serveur/cgi-bin/nph-cgi4d.exe/~-echo
- **Informations sur l'extension** (support technique) :
http://adresse-serveur/cgi-bin/4disapi.dll/~-info
http://adresse-serveur/cgi-bin/nph-cgi4d.exe/~-info

Les informations suivantes sont retournées :

- nom et version de l'extension, par exemple "Script name: 4disapi.dll (6.7.0b1.2)"
- nom et version du serveur appelant l'extension, par exemple "Server software: 4D_WebStar_D/6.7"
- version du protocole HTTP, par exemple "Server protocol: HTTP/1.0"
- version du protocole CGI, par exemple "Gateway interface: CGI/1.1"
- Test du serveur cible (est-il joignable ?) :
http://adresse-serveur/cgi-bin/4disapi.dll/~target
http://adresse-serveur/cgi-bin/nph-cgi4d.exe/~target

La réponse est soit :

- "Good: target server reached." : le serveur cible a répondu (quel que soit le contenu de la réponse).
- soit "Bad: target server not reached." : le serveur est injoignable ou n'a pas répondu.
Dans ce cas, la raison de l'échec peut être :
 - pas de fichier de configuration ;
 - l'adresse ou le port cible est incorrect ;
 - le serveur cible n'est pas en service ;
 - le serveur a traité la requête mais est incapable de répondre.

4D WebSTAR

4D WebSTAR[®] est un des serveurs Web les plus répandus sur la plateforme MacOS. La version 6.7 de 4^e Dimension offre la possibilité d'accéder directement à 4D WebSTAR depuis 4D. Inversement, 4D peut répondre directement à une requête de 4D WebSTAR.

Aucun composant supplémentaire n'est requis — en fait, 4D se comporte comme un CGI pour 4D WebSTAR. Pour configurer cette solution, il suffit de placer un 4D ou un alias de 4D dans le dossier contenant les pages à publier, ou dans le dossier cgi-bin de 4D WebSTAR. Le nom de l'application 4D doit se terminer par ".acgi", par exemple 4D.acgi. Le serveur Web 4D doit être activé préalablement à toute requête et doit répondre sur un port différent de celui de 4D WebSTAR.

Le principe de fonctionnement est le suivant : lorsqu'un URL ou une action du type "/cgi-bin/4d.acgi\$/4daction/proc" est reçue par le serveur 4D WebSTAR, celui-ci envoie un événement spécifique à 4D.acgi via un *Apple Event*. 4D traite alors l'*Apple Event* comme une requête HTTP. Dans notre exemple, la requête HTTP serait "/4daction/proc".

Note Le mode CGI 4D WEBSTAR n'est pas compatible avec le mode contextuel.

Gestion des pages HTML

Outre un nouvel URL spécial, la version 6.7 de 4D comporte plusieurs nouveautés concernant les balises et les variables insérées dans les pages HTML.

URL spécial /4DWEBTEST

Un nouvel URL spécial est disponible dans le serveur Web 4D : /4DWEBTEST. Cet URL est toujours accessible.

Lorsque cet URL est appelé, 4D retourne un fichier texte pur contenant les champs HTTP suivants :

- **Date** : date du jour au format RFC 822
exemple : "Date: Wed, 26 Jan 2000 13:12:50 GMT"
- **Server** : 4D WebStar_D/numéro de version
exemple : "4D WebStar_D/6.7"
- **User-Agent** : nom et version @ adresse IP du client
exemple : "Mozilla/4.08 (Macintosh; I; PPC, Nav) @ 192.193.00.00"

Suppression de la limite des 32000 caractères de texte

Dans les versions précédentes de 4D, l'insertion de variables 4D dans les pages statiques s'effectuait via des variables process 4D de type Texte (débutant par le code ASCII 1). Cette fonctionnalité se heurtait à la limitation inhérente aux variables 4D de type Texte : ces variables n'acceptent pas plus de 32 000 caractères.

Avec la version 6.7 de 4^e Dimension, vous pouvez vous affranchir de cette limite en utilisant des variables de type BLOB. Le principe de fonctionnement est identique à celui des variables Texte : la syntaxe à employer est du type <!--4DVAR leblob-->.

Note Le BLOB doit avoir été généré en mode Texte sans longueur.

Suivant le résultat que vous souhaitez obtenir, vous pouvez ou non insérer un caractère ASCII 1 au début de la variable BLOB. A noter également la possibilité d'utiliser la nouvelle balise 4DHTMLVAR qui insère le contenu de la variable en tant qu'expression HTML (cf. ci-dessous).

Nouvelles balises HTML

La gestion des balises insérées sous forme de commentaires HTML dans les pages envoyées par le serveur Web a été enrichie dans 4D 6.7.

Les nouvelles balises suivantes sont disponibles :

- 4DHTMLVAR, analogue à 4DVAR mais insérant de l'HTML,
- 4DSCRIPT, nouvelle balise se substituant à 4DACTION,
- 4DINCLUDE, permettant d'inclure une page dans une autre,
- 4DIF, 4DELSE et 4DENDIF, permettant d'insérer des conditions dans le code HTML,
- 4DLOOP et 4DENDLOOP, permettant d'effectuer des boucles dans le code HTML,

En outre, les modifications suivantes ont été effectuées :

- les balises 4DVAR et 4DHTMLVAR acceptent désormais les expressions 4D en plus des variables,
- la gestion des comptes-rendus d'erreurs d'analyse a été améliorée.

Utilisation d'expressions 4D dans les commentaires HTML (4DVAR et 4DHTMLVAR)

Il est désormais possible d'insérer des expressions 4D (et non plus seulement des variables) dans les commentaires HTML 4D à l'aide des balises 4DVAR et 4DHTMLVAR. Ainsi, vous pouvez insérer directement le contenu d'un champ (par exemple <!--4DVAR [nomTable]nomChamp-->) ou d'un élément de tableau (par exemple <!--4DVAR tab{1}-->).

La conversion de l'expression obéit aux mêmes règles que la conversion d'une variable. En outre, l'expression doit satisfaire aux règles de syntaxe de 4D.

Bien qu'une expression puisse contenir directement des appels de fonctions 4D, il est déconseillé d'utiliser ce mode de fonctionnement, pour des raisons de localisation. Par exemple le commentaire <!--4DVAR Date du jour-->, bien que correctement interprété avec un 4D français, ne sera pas traité avec un 4D anglais. Il est en de même pour le traitement des nombre réels (car le séparateur décimal peut varier d'un pays à l'autre). Dans ces deux cas, il est préférable d'affecter une variable par programmation.

En cas d'erreur d'évaluation, le texte inséré sera de la forme "`<!--4DVAR mavar--> : ## erreur # code d'erreur`".

-
- Notes*
- Il est possible d'afficher le contenu d'un champ image. En outre (en mode contextuel uniquement), vous pouvez également afficher le contenu d'une variable image.
 - En revanche, dans les deux modes, il n'est pas possible d'afficher le contenu d'un élément de tableau image.
-

Commentaire `<!--4DSCRIPT/meth-->`

Dans les versions précédentes de 4D, un même marqueur, 4D ACTION, pouvait être utilisé soit sous forme d'URL (par exemple `http://monserver/4D ACTION/meth`), soit sous forme de commentaire HTML dans une page statique (`<!--4D ACTION/meth-->`).

Cette diversité d'utilisations pouvant entraîner des confusions, la version 6.7 de 4D comporte une nouvelle balise de commentaire permettant de dissocier les deux usages : 4DSCRIPT.

La balise 4DSCRIPT s'utilise exclusivement sous forme de commentaire HTML (`<!--4DSCRIPT/meth-->`) et produit exactement les mêmes effets que `<!--4D ACTION/meth-->`.

L'instruction 4D ACTION est désormais réservée à une utilisation en tant qu'URL.

-
- Note* Les balises `<!--4D ACTION/meth-->` restent acceptées par 4D 6.7, toutefois nous vous recommandons de prendre l'habitude d'utiliser la nouvelle balise 4DSCRIPT.
-

Commentaire <!--4DHTMLVAR mavar-->

Ce nouveau commentaire permet d'évaluer une variable ou une expression 4D et de l'insérer dans une page en tant qu'expression HTML. En fait, cette balise produit exactement les mêmes effets que la balise <!--4DVAR mavar-->, lorsque mavar débute par le caractère de code ASCII 1.

- ▼ Exemple : voici les résultats de l'insertion de la variable Texte 4D *mavar* à l'aide des balises disponibles.

Valeur de <i>mavar</i>	Balise utilisée	Insertion dans la page Web
mavar:=""	<!--4DVAR mavar-->	
mavar:=Caractere(1)+""	<!--4DVAR mavar-->	
mavar:=""	<!--4DHTMLVAR mavar-->	

Note Les expressions 4D insérées dans les commentaires doivent obéir aux règles de syntaxe de 4D (cf. [paragraphe "Utilisation d'expressions 4D dans les commentaires HTML \(4DVAR et 4DHTMLVAR\)", page 102](#)).

En cas d'erreur d'évaluation, le texte inséré sera de la forme "<!--4DHTMLVAR mavar--> : ## erreur # code d'erreur".

Commentaire <!--4DINCLUDE chemin-->

Ce nouveau commentaire permet d'inclure, dans une page HTML, le corps d'une autre page HTML (désignée par le paramètre chemin). Le corps d'une page HTML désigne ce qui est compris entre les balises <BODY> et </BODY> (les balises elles-mêmes ne sont pas incluses).

Le commentaire <!--4DINCLUDE --> s'avère particulièrement utile en combinaison avec les tests (<!--4DIF-->) ou les boucles (<!--4DLOOP-->). Il est également pratique pour inclure des bannières en fonction d'un critère, ou de manière aléatoire.

Au moment de l'inclusion, quels que soient le mode et l'extension du nom du fichier, 4D analyse la page appelée puis insère son contenu — éventuellement modifié — dans la page d'où provient l'appel 4DINCLUDE.

Note Le placement dans le cache Web d'une page incluse à l'aide du commentaire <!--4DINCLUDE --> répond aux mêmes règles que celles des pages demandées via un URL ou envoyées par la commande ENVOYER FICHIER HTML (cf. manuel *Langage*).

Passez dans chemin le chemin d'accès au document à inclure. Le chemin d'accès est relatif au document en cours d'analyse. Utilisez la barre oblique (/) comme séparateur de dossiers et les deux-points (..) pour remonter d'un niveau hiérarchique (syntaxe HTML).

Le nombre de `<!--4DINCLUDE chemin-->` au sein d'une page n'est pas limité.

Toutefois, les appels à `<!--4DINCLUDE chemin-->` ne peuvent s'effectuer que sur 1 niveau. Cela signifie que par exemple vous ne pouvez pas insérer le commentaire `<!--4DINCLUDE mondoc3.html-->` dans le corps de la page `mondoc2.html`, elle-même appelée par `<!--4DINCLUDE mondoc2-->` inséré dans `mondoc1.html`.

En outre, 4D contrôle que les inclusions ne sont pas récursives.

En cas d'erreur, le texte inséré est de la forme "`<!--4DINCLUDE chemin-->` : Le document ne peut pas être ouvert".

Note En mode contextuel, si une page est insérée dans un formulaire via un marqueur `{mapage.html}` inséré dans une zone de texte statique, les éventuels commentaires 4DINCLUDE qu'elle contient seront ignorés.

▼ Exemples

```
<!--4DINCLUDE souspage.html-->
<!--4DINCLUDE dossier/souspage.html-->
<!--4DINCLUDE ../dossier/souspage.html-->
```

Commentaires `<!--4DIF expression-->` `<!--4DELSE-->` `<!--4DENDIF-->`

Utilisé en conjonction avec les commentaires `<!--4DELSE-->` (optionnel) et `<!--4DENDIF-->`, le commentaire `<!--4DIF expression-->` offre la possibilité d'exécuter du code HTML de manière conditionnelle.

Le paramètre `expression` peut contenir toute expression 4D valide retournant une valeur booléenne. Elle doit figurer entre parenthèses et respecter les règles de syntaxe de 4D.

Note Bien qu'une expression puisse contenir des appels de fonctions 4D, il est déconseillé de le faire pour des raisons de localisation (cf. [paragraphe "Utilisation d'expressions 4D dans les commentaires HTML \(4DVAR et 4DHTMLVAR\)"](#), page 102).

Les blocs `<!--4DIF expression--> ... <!--4DENDIF-->` peuvent être imbriqués sur plusieurs niveaux. Comme dans 4D, chaque `<!--4DIF expression-->` doit avoir un `<!--4DENDIF-->` correspondant.

En cas d'erreur d'évaluation, le texte "`<!--4DIF expression-->` : Une expression booléenne était attendue" est inséré à la place du contenu situé entre `<!--4DIF -->` et `<!--4DENDIF-->`.

De même, s'il n'y a pas autant de `<!--4DENDIF-->` que de `<!--4DIF -->`, le texte "`<!--4DIF expression-->` : 4DENDIF attendu" est inséré à la place du contenu situé entre `<!--4DIF -->` et `<!--4DENDIF-->`.

- ▼ Cet exemple de code inséré dans une page HTML statique affiche un libellé différent en fonction du résultat de l'expression `vnom#""` :

```
<BODY>
...
<!--4DIF (vnom#"")-->
Noms commençant par <!--4DVAR vnom-->.
<!--4DELSE-->
Aucun nom n'a été trouvé.
<!--4DENDIF-->
...
</BODY>
```

Commentaires `<!--4DLOOP condition-->` `<!--4DENDLOOP-->`

Ce commentaire permet de répéter une portion de code HTML tant que la condition est remplie. La portion est délimitée par `<!--4DLOOP-->` et `<!--4DENDLOOP-->`.

Les blocs `<!--4DLOOP condition-->` ... `<!--4DENDLOOP-->` peuvent être imbriqués. Comme dans 4D, chaque `<!--4DLOOP condition-->` doit avoir un `<!--4DENDLOOP-->` correspondant.

Il existe trois types de conditions :

- `<!--4DLOOP [table]-->`
Cette syntaxe effectue une boucle pour chaque enregistrement de la sélection courante de table dans le process en cours. La portion de code HTML située entre les deux commentaires est répétée pour chaque enregistrement de la sélection courante.

Note Lors de l'utilisation de 4DLOOP avec une table, les enregistrements sont chargés en mode Lecture seule.

- ▼ L'exemple de code HTML suivant :

```
<!--4DLOOP [Personnes]-->
<!--4DVAR [Personnes]Nom--> <!--4DVAR [Personnes]Prenom--><BR>
<!--4DENDLOOP-->
```

... peut se traduire en code 4D par :

```
DEBUT SELECTION([Personnes])
Tant que(Non(Fin de selection([Personnes])))
...
  ENREGISTREMENT SUIVANT([Personnes])
Fin tant que
```

■ <!--4DLOOP tableau-->

Cette syntaxe effectue une boucle pour chaque élément du tableau. L'indice courant du tableau est incrémenté à chaque répétition de la portion de code HTML.

Note Il n'est pas possible d'utiliser cette syntaxe avec des tableaux à deux dimensions. Dans ce cas, la solution consiste à combiner une méthode des boucles imbriquées.

▼ L'exemple de code HTML suivant :

```
<!--4DLOOP tab_noms-->
<!--4DVAR tab_noms{tab_noms}--><BR>
<!--4DENDLOOP-->
```

... peut se traduire en code 4D par :

```
Boucle($indice;1;Taille tableau(tab_noms))
  tab_noms:=$indice
...
Fin de boucle
```

■ <!--4DLOOP méthode-->

Cette syntaxe effectue une boucle tant que la méthode retourne Vrai. La méthode admet un paramètre de type Entier long. Elle est appelée une première fois avec la valeur 0 pour permettre une éventuelle phase d'initialisation, puis elle est appelée successivement avec les valeurs 1, 2, 3... tant qu'elle renvoie Vrai.

Pour des raisons de sécurité, la méthode base Sur authentification Web peut être appelée, une seule fois, avant la phase d'initialisation (exécution de la méthode avec 0 comme paramètre). Si l'authentification est confirmée, la phase d'initialisation a lieu.

Note En vue de la compilation de la base, il est impératif de déclarer C_BOOLEEN(\$0) et C_ENTIER LONG(\$1) au sein de la méthode.

▼ L'exemple de code HTML suivant :

```
<!--4DLOOP ma_methode-->
<!--4DVAR var--> <BR>
<!--4DENDLOOP-->
```

... peut se traduire en code 4D par :

```
Si(AuthenticationWebOK)
  Si(ma_methode(0))
    $compteur:=1
    Tant que(ma_methode($compteur))
      ...
      $compteur:=$compteur+1
    Fin tant que
  Fin de si
Fin de si
```

La méthode *ma_methode* pourrait avoir la forme suivante :

```
C_ENTIER LONG($1)
C_BOOLEAN($0)
Si ($1=0)
  `Initialisation
  $0:=Vrai
Sinon
  Si($1<50)
    ...
    var:= ...
    $0:=Vrai
  Sinon
    $0:=Faux `Arrêt de la boucle
  Fin de si
Fin de si
```

En cas d'erreur d'évaluation, le texte "<!--4DLOOP expression--> : descriptif" est inséré à la place du contenu situé entre <!--4DLOOP --> et <!--4DENDLOOP-->.

Le descriptif de l'erreur peut être l'un des suivants :

- Une expression de ce type n'était pas attendue (erreur générique).
- Nom de table invalide (erreur sur le nom de la table).
- Un tableau était attendu (la variable n'est pas un tableau ou est un tableau à deux dimensions).
- La méthode n'existe pas.

- Erreur de syntaxe (lors de l'exécution de la méthode).
- Erreur de privilège (pas de droits suffisants pour accéder à la table ou à la méthode).
- 4DENDLOOP attendu (le nombre de <!--4DENDLOOP--> n'est pas égal à celui de <!--4DLOOP -->).

Notes relatives aux commentaires HTML eux-mêmes

Les marqueurs HTML sont toujours insérés dans les pages Web en tant que commentaires (<!-- --> dans le source HTML). La majorité des éditeurs HTML offrent des facilités pour insérer des commentaires.

Pour rappel, voici les cas dans lesquels 4D analyse les balises contenues dans les pages HTML envoyées aux browsers Web :

Conditions d'envoi	Analyse du contenu des pages envoyées	
	Mode contextuel	Mode sans contexte
Extensions des pages (cas général) : <ul style="list-style-type: none">• .htm, .html, .shtm, .shtml (pages HTML)• .xml, .xsl (pages XML)• .wml (pages WML)	X	X
Appel de la commande ENVOYER FICHIER HTML	X	X
Appel de la commande ENVOYER BLOB HTML (si le blob est du type "text/html")	X	X
Inclusion par le marqueur <!--4DINCLUDE-->	X	X
Inclusion par le marqueur {mapage.htm}	X	-
Pages appelées par des URLs	X	X, sauf pages suffixées ".htm" ou ".html"

En vue d'une exploitation par 4D, un commentaire HTML doit toujours être de la forme <!--4D...-->. Attention toutefois, certains éditeurs HTML ajoutent automatiquement d'autres informations au sein d'un commentaire, ce qui peut entraîner des erreurs d'analyse.

La présence d'autres commentaires HTML (par exemple <!--Debut de liste--> est toutefois possible.

Si un commentaire <!--4D... ne se termine pas par -->, un texte de la forme "<!--4D... : --> attendu" sera inséré et l'analyse sera interrompue à ce niveau (la page sera tout de même envoyée pour indiquer l'erreur).

Comme indiqué précédemment, il est possible d’imbriquer les différents types de commentaires. Voici par exemple une structure HTML parfaitement envisageable :

```
<HTML>
...
<BODY>
<!--4DSCRIPT/PRE_PROCESS-->           (Appel de méthode)
<!--4DIF (mavar=1)-->                 (Si condition)
  <!--4DINCLUDE banner1.html-->       (Insertion d’une sous page)
<!--4DENDIF-->                       (Fin de si)
<!--4DIF (mavar=2)-->
  <!--4DINCLUDE banner2.html-->
<!--4DENDIF-->

<!--4DLOOP [TABLE]-->                 (Boucle sur la sélection
courante)
<!--4DIF ([TABLE]ValNum>10)-->         (Si [TABLE]ValNum>10)
  <!--4DINCLUDE subpage.html-->       (Inclusion d’une sous page)
<!--4DELSE-->                         (Sinon)
  <B>Valeur : <!--4DVAR [TABLE]ValNum--></B><BR>
                                     (Affichage d’un champ)
<!--4DENDLOOP-->                     (Fin de boucle)
</BODY>
</HTML>
```

Optimisation du cache Web

Le rechargement des pages Web a été optimisé dans 4D 6.7, en particulier lorsque l’utilisateur demande à “actualiser” le contenu d’une page.

Dans les versions précédentes de 4D, la page était systématiquement renvoyée dans ce cas. Désormais, 4D compare la date de modification de la page sur disque avec celle indiquée par le navigateur. Si la page n’a pas été modifiée, 4D renvoie la réponse avec le statut “304 - Not Modified”. Cette optimisation a pour but de réduire le trafic réseau.

Le chargement conditionnel (uniquement valable avec la méthode GET) est largement exploité par les serveurs “proxy”.

Connexions sécurisées via SSL

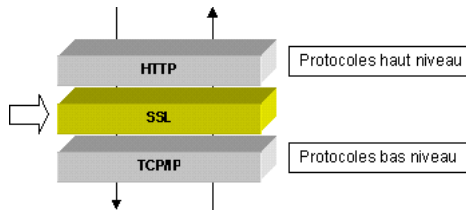
A compter de la version 6.7 de 4D, le serveur Web 4D peut communiquer en mode sécurisé via le protocole SSL (*Secured Socket Layer*).

Qu'est-ce que le protocole SSL ?

Le protocole SSL a pour but de sécuriser les communications entre deux applications — principalement un serveur Web et un browser¹. Ce protocole est largement répandu et compatible avec la plupart des browsers Web.

Au niveau de l'architecture réseau, le protocole SSL s'insère entre la couche TCP/IP (bas niveau) et le protocole de haut niveau HTTP, pour lequel il est principalement destiné.

Architecture réseau utilisant SSL



Le protocole SSL permet de garantir l'identité de l'émetteur et du récepteur, ainsi que la confidentialité et l'intégrité des informations échangées :

- **Identification des intervenants** : l'identité de l'émetteur et du récepteur sont confirmées.
- **Confidentialité des informations échangées** : les données envoyées sont cryptées afin de les rendre inintelligibles pour les tiers non autorisés.
- **Intégrité des informations échangées** : les données reçues n'ont pas été altérées, frauduleusement ou accidentellement.

Les principes de sécurisation utilisés par SSL sont basés sur l'emploi d'un algorithme de cryptage utilisant une paire de clés : une clé privée et une clé publique.

1. Le protocole SSL peut également être utilisé pour sécuriser les connexions client/serveur "classiques" de 4D Server. Pour plus d'informations, reportez-vous au [paragraphe "Connexions sécurisées \(SSL\)"](#), page 23.

La clé privée est utilisée pour crypter les données. Elle est conservée par l'émetteur (le site Web). La clé publique est utilisée pour décrypter les données. Elle est diffusée auprès des récepteurs (les browsers Web), via le certificat. L'emploi du SSL dans le cadre d'Internet requiert en effet l'entremise d'un opérateur de certification tel que, par exemple, Verisign[®]. Moyennant une participation financière du site Web demandeur, cet organisme délivre un certificat, garantissant l'identité du serveur et contenant la clé publique permettant la communication en mode sécurisé.

Note Pour plus d'informations sur les principes généraux de cryptage et d'emploi de clés publiques/clés privées, reportez-vous à la description de la nouvelle commande CRYPTER BLOB.

Installation et activation de SSL dans 4D

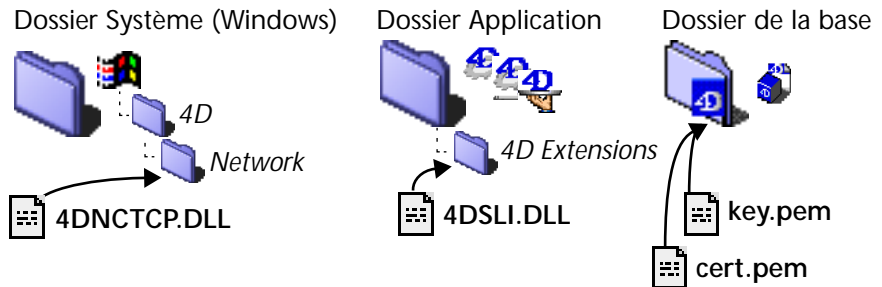
L'intégration du protocole SSL dans 4D a été réalisée au niveau des composants réseau. Plusieurs éléments doivent être présents sur le poste, à différents emplacements :

- **4DNCTCP.DLL** (Windows) : DLL du composant réseau destiné au protocole TCP/IP. Sous Windows, ce fichier doit se trouver dans le dossier 4D\Network du système de la machine (C:\Windows\4D\Network\).
Sous MacOS, les composants réseau sont intégrés aux applications.
- **4DSLI.DLL¹** : interface de la couche sécurisée (*Secured Layer Interface*) dédiée à la gestion du SSL.
Ce fichier doit se trouver dans le dossier [4D Extensions] de l'application 4D qui publie la base sur le Web.
- **key.pem** (serveur Web uniquement) : document contenant la clé de cryptage privée.
Ce fichier doit se trouver dans le dossier de la base.

1. La présence de ce composant est également nécessaire pour l'utilisation des commandes de cryptage des données **CRYPTER BLOB** et **DECRYPTER BLOB**.

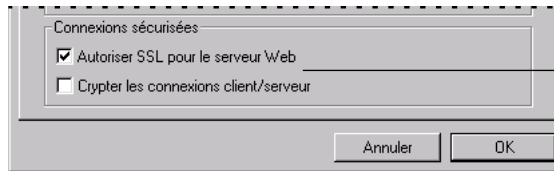
- **cert.pem** (serveur Web uniquement) : document contenant le *certificat* (cf. [paragraphe “Obtenir un certificat SSL”, page 114](#)).
Ce fichier doit se trouver dans le dossier de la base.

Emplacement des fichiers nécessaires au fonctionnement de SSL avec le serveur Web 4D



Une fois ces éléments installés, les connexions au serveur Web ainsi que, éventuellement, les connexions client/serveur, peuvent s’effectuer en mode sécurisé.

Par défaut, les connexions SSL sont activées pour le serveur Web et inactivées pour les connexions client/serveur. Ces paramètres sont accessibles dans la page “Connexions” des Propriétés de la base :



Options d’activation du protocole SSL

Pour plus d’informations sur ce point, reportez-vous au [paragraphe “Connexions sécurisées \(SSL\)”, page 23](#).

Le port TCP réservé aux communications SSL est le 443. Par conséquent, vous ne pouvez pas installer plus d’un serveur Web 4D utilisant SSL par poste. Le port TCP défini dans la page “Serveur Web I” des Propriétés de la base sera utilisé pour les connexions du serveur Web en mode standard.

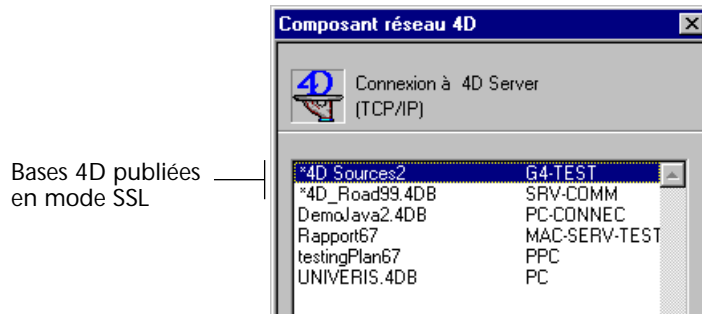
De manière générale, les Propriétés de la base définies pour la gestion du serveur Web 4D (mots de passe, délai avant déconnexion, taille du cache, etc.) restent appliquées, que le serveur fonctionne en mode sécurisé ou non.

Paramétrer SSL en client/serveur

Si vous souhaitez utiliser SSL pour vos connexions client/serveur “classiques” uniquement, seuls les fichiers 4DNCTCP.DLL (sous Windows uniquement) et 4DSLI.DLL sont requis. Ils doivent être installés sur le poste serveur et sur chaque poste client. Il vous suffit de

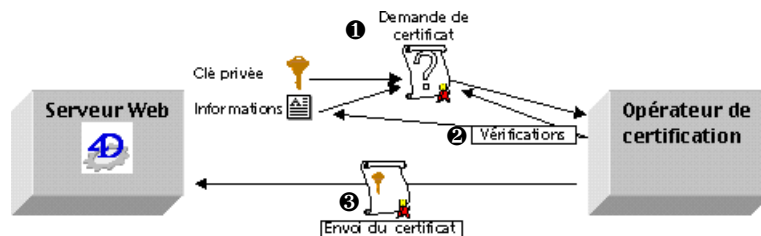
cocher l'option "Crypter les connexions client/serveur" dans les Propriétés de la base. Les mécanismes de génération de clés et d'authentification sont "transparents", ils sont gérés par 4D Server et ne nécessitent pas d'intervention de l'utilisateur.

Un astérisque (*) apparaît devant le nom des bases publiées en mode SSL, dans la boîte de dialogue TCP/IP de connexion à 4D Server. Les bases publiées en mode sécurisé apparaissent donc en tête de la liste :



Obtenir un certificat SSL

La mise en place d'un serveur Web 4D fonctionnant en SSL nécessite un certificat numérique délivré par un opérateur de certification. Ce certificat renferme diverses informations dont la carte d'identité du site ainsi que la clé publique utilisée pour communiquer avec lui. Il est transmis aux browsers Web se connectant au site. Une fois qu'il est accepté¹, la communication en mode sécurisé s'établit.



Le choix de l'autorité de certification dépend de plusieurs facteurs. Plus l'autorité est "connue", plus le nombre de browsers acceptant les certificats qu'elle délivre sera important, mais plus le prix à payer sera élevé. Verisign[®] est une des autorités de certifications les plus utilisées.

1. Pour qu'un browser accepte les certificats d'une autorité de certification, celle-ci doit être répertoriée dans ses Propriétés.

► Pour obtenir un certificat SSL :

- 1 Générez une "clé privée" à l'aide de la commande GENERER CLES CRYPTAGE.

Pour plus d'informations, reportez-vous à la description de la [routine GENERER CLES CRYPTAGE](#), page 57.

Pour des raisons de sécurité, la clé privée ne doit JAMAIS être diffusée sur un réseau. En fait, elle ne doit pas quitter le poste serveur Web. Le fichier Key.pem doit être placé dans le dossier de la structure de la base.

- 2 Etablissez une demande de certificat à l'aide de la commande GENERER DEMANDE CERTIFICAT.

Pour plus d'informations, reportez-vous à la description de la [routine GENERER DEMANDE CERTIFICAT](#), page 59.

- 3 Envoyez la demande de certificat à l'autorité de certification que vous avez choisie.

Pour remplir la demande de certificat, il vous sera peut-être nécessaire de contacter l'autorité de certification. Les autorités de certification vérifient la réalité des informations qui leur ont été transmises.

La demande de certificat est générée dans un BLOB au format PEM (*Privacy Enhanced Mail*). Ce format autorise le copier-coller des clés sous forme de texte et leur envoi par E-mail sans risque d'altération de leur contenu. Vous pouvez donc par exemple sauvegarder le BLOB contenant la demande de certificat dans un document texte (à l'aide de BLOB VERS DOCUMENT), puis l'ouvrir et copier-coller son contenu dans un E-mail ou un formulaire Web destiné à l'autorité de certification.

- 4 Une fois que vous avez reçu votre certificat, créez un fichier texte que vous nommerez "Cert.pem" et copiez dans ce fichier le contenu du certificat.

Vous pouvez recevoir votre certificat sous plusieurs formes (généralement via un E-mail ou un formulaire HTML). Le serveur Web 4D accepte la plupart des formats de texte (MacOS, PC, Linux...) pour les certificats. En revanche, le certificat doit être au format PEM.

- 5 Placez le fichier "cert.pem" dans le dossier contenant la structure de la base.

Le serveur Web peut dès lors fonctionner en mode SSL. La durée de validité d'un certificat varie généralement entre six mois et un an.

Connexions des browsers en SSL

Lorsqu'un browser se connecte à un serveur Web fonctionnant en mode sécurisé, une boîte de dialogue d'alerte le lui signale. Une fois la boîte de dialogue validée, le certificat est envoyé au browser par le serveur Web.




Les deux parties “négocient” alors l’algorithme de cryptage qui va être utilisé pour la connexion. Le serveur Web 4D dispose de plusieurs algorithmes de cryptage symétriques (RC2, RC4, DES...). L’algorithme commun le plus puissant est utilisé.

Le niveau de cryptage autorisé est soumis à la législation en vigueur dans le pays d'utilisation.

Les niveaux de cryptage proposés par le serveur Web 4D dépendent de la version de la bibliothèque système de cryptage utilisée. Par défaut, 4D SA fournit une version “Export” de la bibliothèque, dont les algorithmes symétriques sont limités à 40 bits.

Du côté du browser, deux indications permettent d’identifier la nature sécurisée de la connexion :

- L’URL affiché dans la zone d’adresse du browser débute par “HTTPS:” au lieu de “HTTP:”. Le nom de protocole “https” indique que la connexion s’effectue en mode SSL.
- Un symbole représentant un cadenas fermé s’affiche en bas de la fenêtre du navigateur . L’utilisateur peut double-cliquer sur ce cadenas pour obtenir des informations sur la connexion (certificat, etc.).

Mise à jour des serveurs Web existants

L’utilisation de SSL dans le serveur Web 4D ne nécessite pas de configuration système particulière.

Toutefois, vous devez tenir compte du fait qu’un serveur Web SSL peut également fonctionner en mode non sécurisé. Le changement de mode de connexion peut s’effectuer si le browser en fait la demande (il suffit par exemple à l’utilisateur, dans la zone d’URL du browser, de remplacer “HTTPS” par “HTTP”). Il revient au développeur d’interdire ou de rediriger les requêtes effectuées en mode non sécurisé — la nouvelle [routine Connexion Web securisee](#), page 63, permet de connaître le mode de connexion courant.

De même, dans le cadre de la mise à jour en version 6.7 SSL d'un serveur Web 4D existant, veillez à ce que les URLs de navigation complets placés dans les pages et référant d'autres pages du même site débutent bien par "HTTPS", sinon ils provoqueront le basculement de la connexion en mode non sécurisé.

Note Le sécurité des fichiers du serveur Web 4D a été renforcée par la définition d'un dossier racine HTML par défaut pour les nouvelles bases créées en version 6.7 et suivantes. Pour plus d'informations sur ce point, reportez-vous au [paragraphe "Racine HTML par défaut : DossierWeb"](#), page 26.

5

Optimisations

La version 6.7 de 4D inclut diverses optimisations contribuant à améliorer les performances et le confort d'utilisation des applications 4D.

Editeur de commentaires

La taille maximale des commentaires associés aux objets a été portée à 32 700 caractères.

En outre, la gestion des fonctions d'annulation et la vitesse de défilement du texte ont été améliorés.

Index, recherches et tris

Recherches par contenu

La recherche par contenu effectuée sur un champ (par exemple [Clients]Nom = "@anne@") tire désormais parti de l'index, si le champ est indexé. Ce type de recherche était auparavant toujours séquentiel.

Tris

Les tris sur des champs indexés sont désormais exécutés plus rapidement dans les cas suivants :

- le nombre d'enregistrements appartenant à la sélection courante est faible par rapport au nombre total d'enregistrements.
- un tri décroissant est effectué sur un champ indexé (dans les versions précédentes de 4D, les tris décroissants pouvaient s'exécuter de manière nettement plus lente que les tris croissants).

Nombre de clés d'index

Dans les versions précédentes de 4D, le nombre de clés par index était limité à 8 millions. Cette limite a été repoussée avec 4D 6.7. Un index créé avec cette version peut désormais contenir 16 millions de clés par index.

Note Il ne faut pas confondre les *clés d'index* avec les *enregistrements indexés*. En effet, les clés d'index concernent à la fois les enregistrements et les sous-enregistrements.

Les bases créées avec une version antérieure de 4D ne sont pas converties par 4D 6.7, donc leurs index seront toujours limités à 8 millions de clés. Toutefois, pour bénéficier de cette optimisation il suffit de réindexer les champs avec 4D 6.7.

Index

Chiffres

4D

- Dossier 4D 9
- 4D Extensions 12
- 4D Web Assistant 14
- 4D WebSTAR 100
- 4DELSE (nouvelle balise HTML) 105
- 4DENDIF (nouvelle balise HTML) 105
- 4DENDLOOP (nouvelle balise HTML) 106
- 4DHTMLVAR (nouvelle balise HTML) 104
- 4DIF (nouvelle balise HTML) 105
- 4DINCLUDE (nouvelle balise HTML) 104
- 4DISAPI.DLL 96
 - Installation 98
 - Utilisation 99
- 4DLOOP (nouvelle balise HTML) 106
- 4DNCTCP.DLL 112
- 4DSCRIPT (nouvelle balise HTML) 103
- 4DSLI.DLL 112
- 4DWEBTEST (URL spécial) 101

A

ACI

- Changement de nom 9
- Dossier ACI 9
- Activation de SSL 112
- Adresse IP d'écoute (Propriétés de la base) 25
- Affichage de la page 0 (raccourci) 17
- Appel au système (Propriétés de la base) 22
- Automatique (Interface de plate-forme) 19
- Autoriser SSL pour le serveur Web (Propriétés de la base) 23

B

Bases 4D

- Transport direct 12
- Bases 4D 6.0.x et 6.5.x 8
- BLOB VERS IMAGE (Nouvelle commande) 53

C

- Cache Web (optimisation) 110
- cert.pem 113
- Certificat SSL 114
- CGI
 - Erreurs 93
 - Informations à destination des développeurs de CGI 94
 - Installation sur le serveur Web 4D 91
 - Interaction avec le serveur Web 4D 93
 - Types de CGI 92
 - Utiliser des CGI 91
- CHANGER PROPRIETES LISTE (Commande modifiée) 70
- CHARGER ET COMPRESSER IMAGE (Commande modifiée) 71
- Clés privées et publiques 58
- Commandes modifiées 70
 - CHANGER PROPRIETES LISTE 70
 - CHARGER ET COMPRESSER IMAGE 71
 - COMPRESSER FICHIER IMAGE 71
 - COMPRESSER IMAGE 70
 - CREER ENSEMBLE SUR TABLEAU 71
 - CREER SELECTION SUR TABLEAU 71
 - DEPLACER OBJET 72
 - Dossier 4D 72
 - Dossier ACI 72
 - Dossier systeme 72
 - FIXER ENTETE HTTP 73
 - FIXER INTERFACE 74
 - FIXER PARAMETRE BASE 75
 - IMAGE VERS GIF 83
 - LIRE IMAGE DANS BIBLIOTHEQUE 80
 - Lire interface 81
 - Lire parametre base 81
 - Lire paramètre base 81
 - LIRE PROPRIETES CHAMP 82
 - LIRE PROPRIETES LISTE 83
 - OUVRIR URL WEB 83
 - PICT VERS GIF 83
 - PROPRIETES CHAMP 82
 - SIECLE PAR DEFAULT 84
 - SUPPRIMER IMAGE DANS BIBLIOTHEQUE 84
 - TABLEAU ENTIER LONG SUR SELECTION 85

- Commentaires (optimisation de l'éditeur) 119
- Commentaires HTML 109
- Compatibilité 8
 - Appels au système 23
 - DossierWeb 26
 - générale 8
 - Interface de plate-forme 19
 - Mode de conversion Web 89
 - SSL et serveurs Web existants 116
- Compatibilité avec les technologies Internet 88
- Composants (Explorateur) 28
- Composants 4D
 - Attributs 29
 - Installer l'assistant Web de 4D 14
 - Types d'objets 28
 - Utilisation 27
 - Visualisation 27
- COMPRESSER FICHIER IMAGE (commandes modifiées) 71
- COMPRESSER IMAGE (Commande modifiée) 70
- Connexion Web securisee (Nouvelle commande) 63
- Connexions en SSL 116
- Connexions sécurisées
 - Propriétés de la base 23
 - SSL 111
- Constante Fenêtre palette 85
- Conversion des bases 8
- Conversion des fichiers 4D 8
- CREER ALIAS (Nouvelle commande) 43
- CREER ENSEMBLE SUR TABLEAU (Commande modifiée) 71
- CREER IMAGETTE (Nouvelle commande) 54
- CREER SELECTION SUR TABLEAU (Commande modifiée) 71
- Cryptage
 - Clés 58
 - Exemples 34
 - Optimisation 34
- CRYPTER BLOB (Nouvelle commande) 32
- Crypter les connexions client-serveur (Propriétés de la base) 24
- CSS1 88

D

- DECRYPTER BLOB (Nouvelle commande) 38
- DEPLACER OBJET (Commande modifiée) 72
- Dossier 4D 9
- Dossier 4D (Commande modifiée) 72

- Dossier 4D Extensions 12
- Dossier ACI 9
- Dossier ACI (Commande modifiée) 72
- Dossier systeme (Commande modifiée) 72
- DossierWeb (Racine HTML par défaut) 26

E

- ECRIRE FICHIER IMAGE (Nouvelle commande) 48
- Editeur de commentaires (Optimisation) 119
- ENVOYER TEXTE HTML (Nouvelle commande) 64
- Explorateur
 - Page Composants 28
- Explorateur d'exécution 21
 - Nombre de requêtes HTTP 21
 - Occupation du cache Web 21
 - Temps d'activité du serveur Web 21
- Expressions 4D dans les pages HTML 102

F

- Fenêtre palette (Constante modifiée) 85
- Feuilles de style 17
- Feuilles de style en cascade (HTML) 88
- Fichiers 4D
 - Compatibilité 8
 - Gestion 12
 - Installation 12
 - Nouvelle architecture 12
- FIXER ENTETE HTTP (Commande modifiée) 73
- FIXER INTERFACE (Commande modifiée) 74
- FIXER PARAMETRE BASE (Commande modifiée) 75
- Formulaire hérité 16
- Formulaires 15
 - Héritage 15
 - Objets 20
 - Orientation des onglets (MacOS) 20
 - Variables non saisissables et clics 20
 - Zones de défilement transparentes 20

G

- GENERER CLES CRYPTAGE (Nouvelle commande) 57
- GENERER DEMANDE CERTIFICAT (Nouvelle commande) 59
- Gestion des pages HTML 101

H

Héritage de formulaire 15

HTML

Gestion 101

Support de HTML 4.0 88

Traitement des commentaires 109

HTTPS 116

I

IMAGE VERS BLOB (Nouvelle commande) 52

IMAGE VERS GIF (Commande modifiée) 83

Index (Optimisation) 119

Installation de 4D Web Assistant 14

Installation de SSL 112

Installer l'assistant Web de 4D (option) 14

Interface de plate-forme 18

Automatique 19

La plus adaptée 19

Mac OS 7 18

Thème Mac 18

Introduction 7

K

key.pem 112

L

La plus adaptée (Interface de plate-forme) 19

Langage 31

Commandes modifiées 70

Nouvelles commandes 32

LIRE ENTETE HTTP (Nouvelle commande) 65

LIRE FICHER IMAGE (Nouvelle commande) 49

LIRE ICONE DOCUMENT (Nouvelle commande) 45

Lire ID ressource composant (Nouvelle commande) 62

LIRE IMAGE DANS BIBLIOTHEQUE (Commande modifiée) 80

LIRE INFORMATIONS SERIALISATION (Nouvelle commande) 46

Lire interface (Commande modifiée) 81

Lire parametre base (Commande modifiée) 81

LIRE PROPRIETES CHAMP (Commande modifiée) 82

LIRE PROPRIETES LIEN (Nouvelle commande) 40

LIRE PROPRIETES LISTE (Commande modifiée) 83

LIRE PROPRIETES SAISIE CHAMP (Nouvelle com-

mande) 39

LIRE PROPRIETES TABLE (Nouvelle commande) 42

LIRE VARIABLES FORMULAIRE WEB (Nouvelle commande) 68

Liste des propriétés (Feuilles de style) 17

LISTE TYPES IMAGES (Nouvelle commande) 50

M

Mac OS 7 (Interface de plate-forme) 18

Mode de conversion Web 89

Mode Structure 11

N

Nom methode courante (Nouvelle commande) 56

Nombre de clés d'index (Optimisation) 120

Nombre de requêtes HTTP (Explorateur d'exécution) 21

Nouvelles balises HTML 102

4DELSE 105

4DENDIF 105

4DENDLOOP 106

4DHTMLVAR 104

4DIF 105

4DINCLUDE 104

4DLOOP 106

4DSCRIPT 103

Nouvelles commandes

BLOB VERS IMAGE 53

Connexion Web securisee 63

CREER ALIAS 43

CREER IMAGETTE 54

CRYPTER BLOB 32

DECRYPTER BLOB 38

ECRIRE FICHER IMAGE 48

ENVOYER TEXTE HTML 64

GENERER CLES CRYPTAGE 57

GENERER DEMANDE CERTIFICAT 59

IMAGE VERS BLOB 52

LIRE ENTETE HTTP 65

LIRE FICHER IMAGE 49

LIRE ICONE DOCUMENT 45

Lire ID ressource composant 62

LIRE INFORMATIONS SERIALISATION 46

LIRE PROPRIETES LIEN 40

LIRE PROPRIETES SAISIE CHAMP 39

LIRE PROPRIETES TABLE 42

LIRE VARIABLES FORMULAIRE WEB 68

- LISTE TYPES IMAGES 50
- Nom methode courante 56
- RESOUDRE ALIAS 44
- SUPPRIMER DOSSIER 45
- NPH-CGI4D.EXE 96
 - Installation 98
 - Utilisation 99

O

- Objets de formulaires 20
- Obtenir un certificat SSL 114
- Occupation du cache Web (Explorateur d'exécution) 21
- Onglets (orientation) 20
- Optimisations 119
 - Cache Web 110
 - Editeur de commentaires 119
 - Index 119
 - Nombre de clés d'index 120
 - Recherches par contenu 119
 - Tris 119
- Orientation des onglets (MacOS) 20
- OUVRIR URL WEB (Commande modifiée) 83

P

- Page 0 (raccourci d'affichage) 17
- Pages HTML
 - Gestion 101
 - Insertion d'éléments de tableaux 102
 - Insertion d'expressions 4D 102
 - Insertion de variables BLOB 101
 - Insertion du contenu d'un champ 102
- PICT VERS GIF (Commande modifiée) 83
- Plan du manuel 7
- Plate-forme
 - Interface 18
- Privé (attribut de composant) 29
- Process Web simultanés maxi (Propriétés de la base) 24
- PROPRIETES CHAMP (Commande modifiée) 82
- Propriétés de la base 22
 - Adresse IP d'écoute 25
 - Appel au système 22
 - Autoriser SSL pour le serveur Web 23
 - Connexions sécurisées (SSL) 23
 - Crypter les connexions client-serveur 24
 - Process Web simultanés maxi 24

- Racine HTML par défaut 26
- Protégé (attribut de composant) 29
- Protocole sécurisé (Nouveau thème) 57
- Protocole SSL, voir SSL 111
- Public (attribut de composant) 29

Q

- QuickTime 4 51

R

- Raccourci d'affichage de la page 0 17
- Racine HTML par défaut
 - DossierWeb 26
- Racine HTML par défaut (Propriétés de la base) 26
 - DossierWeb 26
- Recherches par contenu (Optimisation) 119
- Réserve de process Web 24
- RESOUDRE ALIAS (Nouvelle commande) 44
- RSA (SSL) 58

S

- Serveur Web 87
 - CGI 91
 - Interface avec d'autres serveurs HTTP 96
 - Utiliser des CGI 91
- SIECLE PAR DEFAUT (Commande modifiée) 84
- SSL 111
 - Activation 112
 - Connexion des browsers 116
 - Définition 111
 - Exemples de cryptage 34
 - Installation 112
 - Obtenir un certificat 114
 - Paramétrage en client-serveur 113
 - Propriétés de la base 23
- Structure 11
- SUPPRIMER DOSSIER (Nouvelle commande) 45
- SUPPRIMER IMAGE DANS BIBLIOTHEQUE (Commande modifiée) 84
- Sur clic souris (Variables non saisissables) 20
- Sur double clic souris (Variables non saisissables) 20

T

- TABEAU ENTIER LONG SUR SELECTION (Commande modifiée) 85

Temps d'activité du serveur Web (Explorateur d'exécution) 21

Thème Mac

 Nouvelle constante 74

 Onglets 21

Thème Mac (Interface de plate-forme) 18

Transport des bases 4D 12

Tris (Optimisation) 119

Types de CGI 92

Types de fenêtres 85

U

URL spécial /4DWEBTEST 101

Utilisation des composants 4D 27

V

Variables BLOB dans des pages statiques 101

Variables non-saisissables (Gestion des clics) 20

Visualisation des composants 4D 27

W

WAP 88

WebSTAR et 4D Web Server 101

WML 88

X

XML 90

Z

Zones de défilement transparentes 20

