Programmation du Terminal radio

Table des matières

Worth Data Inc.
21-23 Fenian Street
Dublin 2
Irlande
Tél.: 0800 393 213 ou +353 16 61 45 66
Fax: +353 16 61 46 22
E-mail : wortheur@iol.ie
http://www.codesbarres.com

Mars 2002

Fonctionnement du système

Le terminal radio possède un écran à cristaux liquides de 6x24 (six lignes, vingt quatre caractères par ligne) et accepte jusqu'à 99 messages vocaux qui peuvent être activés par le programme de l'hôte. Les messages du programme de l'ordinateur hôte sont transmis vers le port série auquel la station de base visée est connectée. Une station de base peut contrôler jusqu'à seize terminaux radio. Par conséquent, le programme opérateur de l'hôte doit s'adresser au terminal radio approprié en utilisant son caractère ID. Quand l'hôte recevra un message en provenance de la station de base, les données comprendront le caractère ID du terminal (vrai en bidirectionnel, faux en unidirectionnel).

Les programmes peuvent être écrits dans n'importe quel langage qui permet l'accès au port série (lecture et écriture), quelle que soit la plate-forme. Une seule Station de base est autorisée par port série ; la connexion de stations de base multiples en cascade ou en guirlande à un seul port série n’est pas prise en charge. Pour faciliter votre programmation, nous fournissons des contrôles drop-in ActiveX ainsi que des contrôles TCP/IP. La programmation du Terminal radio n’a jamais été aussi aisée.

Le Terminal radio fonctionne de deux manières principales :

- Communication unidirectionnelle : tout transfert de données est initié par le Terminal radio. La station de base se contente de confirmer la réception des données en les renvoyant au Terminal. L'ordinateur hôte ne dialogue pas avec la station de base ou le terminal. Il sert uniquement à la réception des données entrantes, envoyées par la base via le port série, puis à leur traitement.

- Communication bidirectionnelle : les messages du programme de l'utilisateur hôte sont envoyés à la station de base via le port série, puis de la base au Terminal radio. Le terminal répond à la base avec les données et son identificateur (ID terminal). Les données sont ensuite transmises depuis la base vers l'ordinateur hôte où elles sont traitées. La commande suivante peut être envoyée. Chaque Terminal radio possède un identificateur unique qui permet à une seule station de base de contrôler jusqu'à 16 terminaux. Un dialogue bidirectionnel est établi lorsqu'un terminal se connecte à un réseau à fréquence radio. L'application de l'ordinateur hôte attend le signal de connexion du terminal, puis elle commence son traitement en envoyant la première invite au terminal via la station de base.

- Si le terminal ne reçoit aucune invite en provenance de l'hôte, il se met en "veille" et s'active périodiquement pour vérifier la présence de messages en attente. Ce processus permet une économie d'énergie et une réduction du trafic radio.

Contrairement au mode unidirectionnel, le mode bidirectionnel nécessite une programmation pour la communication avec le terminal. Nous avons essayé de faciliter la communication entre la station de base et le programmeur. Aucun protocole ou "hand-shaking" n'est requis. Ce type de communication fonctionne correctement lorsque la base est située à quelques mètres du port série auquel elle est connectée. Si vous placez votre station de base à une distance supérieure, utilisez un câble gainé et relié à la terre , des débits plus faibles et si possible des gestionnaires de ligne dans les environnements bruyants.

Avant de commencer

Avant de commencer la programmation, vous devez prendre en compte les facteurs suivants dans le processus de planification :

- **Planification des pannes** : elles comprennent les pannes matériel, les pannes logiciel ainsi que les erreurs commises par l'opérateur. Pour créer une application efficace, vous devez anticiper les différentes pannes système potentielles.

- **Recherche de toutes les erreurs** : assurez-vous que votre programme détecte toutes les conditions d'erreur possibles et susceptibles d'être renvoyées par la station de base, parmi lesquelles :

  - erreur de séquence détectée ;
  - commande illégale détectée ;
  - station de base initialisée ;
  - adressage du terminal non connecté.

La programmation de ces conditions d'erreur est souvent négligée. Même si vous pensez que votre code est infaillible, nous vous recommandons de consulter les informations fournies par la base.

- **Analyse attentive des chaînes retournées** : n'anticipez pas la réponse de la base à votre programme et concentrez-vous sur la recherche d'une chaîne partielle telle que l'ID seul. Analysez la chaîne retournée dans

son intégralité et assurez-vous d'avoir envisagé toutes les possibilités. Les erreurs sont fréquentes en effectuant cette tâche.

- **Planification d'une extension** : même si votre structure de départ est petite (1 base, 1 terminal), essayez de créer une application qui vous permettra de vous agrandir facilement et d'ajouter du matériel, en particulier des terminaux.

- **Evaluation du site** : le test d'un site ne nécessite aucune application active. Si vous connaissez vos besoins en termes de station de base et de relais, ce test vous fera gagner du temps lors de la création de votre programme.

- **Utilisation des programmes de démonstration** : les programmes de démonstration vous permettent de découvrir le fonctionnement du système et d'anticiper les problèmes qui peuvent l'affecter. Ces programmes servent également à tester le délai de réponse.

Contrôles ActiveX de PromptCOM

Les composants drop-in (glisser-déplacer) s'ajoutent à vos outils de programmation.

Il existe plusieurs technologies d'implémentation des composants drop-in, telles que VBX (pour Visual basic), VCL (pour Delphi et C Builder) et COM (ActiveX). Seules les technologies ActiveX sont compatibles avec la plupart des environnements de développement.

PromptCOM/ActiveX est un drop in dans un composant COM qui permet à un programmeur d'ajouter facilement l'envoi d'invites et la réception de données en provenance du terminal radio via la station de base radio.
Il est compatible avec Visual Basic, Visual C++, Delphi et la plupart des plate-formes de développement en 32-bits. Pour obtenir des instructions d'installation, consultez le fichier d'aide.

Programmation
Avant de réaliser des appels de méthode, suivez la procédure ci-dessous :

- Configurez les propriétés du port de communication (nom du périphérique, vitesse, parité, bits, contrôle du flux) selon vos besoins. Assurez-vous que le port est fermé (appel **CloseDevice**) avant de modifier ses paramètres.

- Appelez la méthode **OpenDevice**. Cette opération active le port de communication utilisé par cette instance du contrôle **WDterm**.

- Définissez la propriété **ActiveTerminal** (terminal actif) pour identifier le terminal que vous souhaitez utiliser. Vous pouvez changer la propriété **ActiveTerminal** à n'importe quel moment pour rediriger les commandes vers les terminaux appropriés.

Test de qualité de la communication

Implémentez un gestionnaire d'événement pour **OnTermBaseRegister** qui provoque un signal sonore ou l'affichage d'un message lorsqu'il est appelé. Si la communication fonctionne normalement entre l'hôte et la station de base, votre gestionnaire d'événement s'exécutera dès l'activation de votre programme et la mise en marche d'une station de base connectée.

Stations de base multiples

Dans les installations où plusieurs stations de base sont connectées à un seul hôte, appelées "channels" (canaux) dans PromptCOM/DLL, il suffit d'ajouter à votre application un contrôle **WDterm** pour chaque station de base.

Suivi de terminal

Comme vous disposez d'une série de gestionnaires d'événements pour chaque station de base, vous devez élaborer un plan qui vous permettra d'assurer le suivi de chaque terminal (jusqu'à 16 terminaux par station de base) dans sa séquence de transactions. Vous pouvez par exemple utiliser une variable "state" ("état", stockée dans un tableau) pour chaque terminal. Testez cette variable pour déterminer l'invite suivante de n'importe quel terminal. Pour plus d'informations, reportez-vous aux échantillons.

L'"état de connexion" de chaque terminal doit être suivi avec la plus grande attention. Chaque événement **SignOut** (déconnexion) doit être associé à un événement **SignIn** (connexion) et aucun terminal ne doit être autorisé à se connecter deux fois sans événement **SignOut** préalable. Lorsque plusieurs événements **SignIn** proviennent d'un terminal sans événement **SignOut**, cela indique l'un des problèmes suivants :

1) Un terminal hors de portée est mis en marche avant un retour dans la zone de portée
2) Deux terminaux ou plus utilisent le même identificateur (conflit d'ID).

Concepts

Lorsque vous utilisez des composants drop-in dans votre programme, vous suivez le paradigme standard de programmation orientée-objet qui implémente la fonctionnalité du composant drop-in à l'aide de propriétés, de méthodes et d'événements.

Les **propriétés** correspondent aux diverses variables de configuration utilisées par le composant drop-in. Le paramètre *ComDeviceName* en est un exemple.

Les **méthodes** sont des appels de fonction qui fournissent les commandes et les fonctionnalités d'accès du composant drop-in. Vous pouvez par exemple envoyer une commande Input (entrée) au terminal.

Les **événements** sont des définitions de fonction placées dans le code source de votre application. Dans ce dernier, ces définitions sont appelées Event Handlers (gestionnaires d'événements). La structure de base du code source du gestionnaire d'événements est générée automatiquement. Le code du gestionnaire d'événement est appelé (ou lancé) par le composant drop-in lorsqu'un événement spécifique se produit ; par exemple, un terminal renvoie des données et l'événement *OnTermData* est lancé.

Les détails relatifs à l'accès aux propriétés, aux méthodes et aux événements varient selon les plate-formes de développement. Le mode de fonctionnement détaillé des plate-formes les plus connues est illustré dans les échantillons du CD-ROM des Utilitaires ou téléchargeables à partir de notre site :

http://www.barcodehq.com/wdterminal.exe

Propriétés

Les propriétés sont les diverses variables de configuration utilisées par le contrôle **WDterm**. Elles peuvent être assignées directement dans votre

application (par exemple, "**WDterm.ActiveTerminal = 5**") et définies dans l'explorateur d'objets de votre environnement de développement.

Mise en garde : Le port série doit être fermé avant de modifier les propriétés, à l'exception des propriétés ActiveTerminal et Quiet. Utilisez la méthode CloseDevice (fermeture de périphérique) avant de définir les propriétés, puis ouvrez de nouveau le port série à l'aide de OpenDevice (ouverture de périphérique).

Remarque : pour le contrôle **WDterm**, votre environnement de développement peut contenir des propriétés supplémentaires et absentes de cette liste. Ceci est normal et vous pouvez ignorer ces propriétés.

ActiveTerminal
Valeurs possibles : 0 à 15
Définition : Identificateur (de 0 à 15) du terminal vers lequel des instructions d'appel de méthode sont dirigées.

ComDeviceName
Valeurs possibles : COM1 à COM16
Définition : Port série que cette instance du contrôle va utiliser. Si vous possédez plus d'une station de base, drop in un autre contrôle WDterm et définissez son ComDeviceName pour vos autres ports de communication.

ComBaudValue
Valeurs possibles : 300, 600, 1200, 2400, 4800, 9600, 19200
Définition : Il s'agit du paramètre de vitesse du port série qui doit être identique à celui de la station de base.

ComParity
Valeurs possibles : Aucune, paire, impaire.
Définition : Il s'agit du paramètre du port série qui doit être identique à celui de la station de base.
WDterm peut accepter d'autres paramètres mais ceux indiqués ci-dessus sont les seuls à être compatibles avec les stations de base de la version actuelle.

ComDataBits
Valeurs possibles : 7, 8
Définition : Il s'agit du paramètre du port série qui doit être identique à celui de la station de base. WDterm peut accepter d'autres paramètres mais ceux indiqués ci-dessus sont les seuls à être compatibles avec la version actuelle des stations de base.

ComStopBits
Valeurs possibles : 1, 2.
Définition : Il s'agit du paramètre du port série qui doit être identique à celui de la station de base. WDterm peut accepter d'autres paramètres mais ceux indiqués ci-dessus sont les seuls à être compatibles avec la version actuelle des stations de base.

Quiet
Valeurs possibles : True, False.
Définition : Si le paramètre Quiet est défini à True, tous les messages d'état et d'erreur générés par WDterm sont supprimés.

Méthodes

Les méthodes sont des commandes que vous fournissez au contrôle WDterm. Toutes les commandes "Inputxxx" indiquent au terminal d'attendre que l'opérateur entre des données.

Remarque : pour le contrôle WDterm, votre environnement de développement peut indiquer d'autres méthodes qui ne figurent pas dans cette liste. Ceci est normal et vous pouvez ignorer ces propriétés.

Mise en garde : au démarrage de votre application, le port série est "fermé". Vous devez appeler **OpenDevice** pour que les autres appels de méthode puissent fonctionner.

A l'exception de la méthode **ReInitAll**, toutes les méthodes utilisent les propriétés du terminal actif pour identifier le terminal à utiliser.

OpenDevice

Fonction : Cette méthode ouvre le port de communication (port série). La méthode *OpenDevice* doit être appelée avant les méthodes décrites ci-dessous. Assurez-vous de définir toutes les propriétés requises avant d'appeler cette méthode (à l'exception de *ActiveTerminal* et Quiet).

CloseDevice

Fonction : Cette méthode ferme le port de communication (port série). La méthode *CloseDevice* doit être appelée avant de modifier les méthodes décrites ci-dessous (à l'exception de *ActiveTerminal* et Quiet). Au démarrage de votre application, le port série est "fermé". Vous devez appeler *OpenDevice* pour que les autres appels de méthode puissent fonctionner.

InputAny

Paramètres : line, position, prompt, shifted, timestamped

Fonction : Cette méthode instruit le terminal actif (terminal actif) d'afficher l'invite au bon endroit, puis d'attendre les données saisies sur le clavier du terminal ou à l'aide du scanner. Si le paramètre "shifted" (verrouillage majuscules) est défini à True, le terminal démarre en mode shifted. Le paramètre "timestamped" (heure) ajoute un préfixe (au format hhmmss) aux données retournées.

InputKeyBd

Paramètres : line, position, prompt, shifted, timestamped

Fonction : Cette méthode instruit le terminal actif d'afficher l'invite au bon endroit, puis d'attendre la saisie de données à l'aide clavier du terminal. Si le paramètre "shifted" (verrouillage majuscules) est défini à True, le terminal démarre en mode shifted. Le paramètre "timestamped" (heure) ajoute un préfixe (au format hhmmss) aux données retournées.

InputScanner

Paramètres : line, position, prompt, allowbreakout, timestamped

Fonction : Cette méthode instruit le terminal actif d'afficher l'invite au bon endroit, puis d'attendre les données saisies uniquement à l'aide du scanner du terminal. Définir la valeur allowbreakout à True permet à l'utilisateur de désactiver le mode scanner seul en appuyant sur la touche End du terminal. Un termID+CR sera envoyé à l'hôte.

InputYesNo

Paramètres : line, position, prompt

Fonction : Cette méthode instruit le terminal actif d'afficher l'invite au bon endroit, puis d'attendre l'entrée de Yes (touche Enter ou C) ou de No (Touche 0 ou B) à l'aide clavier du terminal. Remarque : les touches C et B sont utilisées pour faciliter la saisie clavier lorsque la lecture avec le laser intégré est active.

InputPassword

Paramètres : line, position, prompt, shifted

Fonction : Cette méthode instruit le terminal actif d'afficher l'invite au bon endroit, puis d'attendre la saisie de données à l'aide clavier du terminal. Les données saisies ne s'affichent pas sur le terminal.

InputSerial
Paramètres : line, position, prompt
Fonction : Cette méthode instruit le terminal actif d'afficher l'invite au bon endroit, puis d'attendre la réception des données via le port série du terminal. L'attente en entrée série peut être évitée en appuyant sur la touche Enter du terminal. Celui-ci enverra une chaîne de données vide à l'hôte (lance le gestionnaire d'événement *OnTermData*).

OutputSerial
Paramètre : data
Fonction : Cette méthode instruit le terminal actif d'envoyer des données au port série du terminal. Les données ne doivent pas comporter plus de 248 caractères pour chaque appel vers *OutputSerial*. Si vous envoyez des données à une imprimante connectée au terminal, assurez-vous de définir le paramètre Protocole du terminal radio à XON/XOFF. Pour plus d'informations, consultez le manuel du terminal radio.

SendDisplay
Paramètres : line, position, prompt
Fonction : Cette méthode instruit le terminal actif d'afficher l'invite au bon endroit. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

ClearScreen
Fonction : Cette méthode instruit le terminal actif d'effacer l'affichage. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

ClearLine
Paramètre : line
Fonction : Cette méthode instruit le terminal actif d'effacer des lignes spécifiques de l'affichage. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

SendDate
Paramètre : line
Fonction : Cette méthode instruit le terminal actif d'afficher la date et l'heure au bon endroit. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

Beep
Paramètre : count
Fonction : Cette méthode instruit le terminal actif de déclencher un signal sonore (en nombre de bips). Le décompte peut avoir une valeur comprise entre 1 et 9. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

PlayVoice
Paramètre : msgnum
Fonction : Cette méthode instruit le terminal actif de diffuser le message vocal "*msgnum*". *Msgnum* peut avoir une valeur comprise entre 1 et 99 et doit être suivi d'un appel de méthode Input (entrée) pour être activé.

ReInit
Fonction : Cette méthode instruit le terminal actif de se réinitialiser. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

Remarque : Le message "base réinitialisée" s'affichera sur les terminaux des stations de base qui utilisent des versions d'EEPROM antérieures à 9079. Seul le terminal a été réinitialisé. Les dernières stations de base affichent le message "Buffer Reinitialized..." pour indiquer la rénitialisation d'un seul terminal.

ReInitAll
Fonction : Cette méthode instruit tous les terminaux connectés de se réinitialiser.

Evénements

Les événements **WDterm** requièrent des conditions spécifiques pour se déclencher. Lorsqu'un événement est lancé, une fonction de gestionnaire d'événement de votre application est appelée. Bien que la façon dont les événements se produisent varie d'un environnement de programmation à un autre, les structures de base des codes sources des divers gestionnaires d'événement sont générés automatiquement et insérés dans votre code source. Pour plus d'informations, reportez-vous aux exemples.

Chaque événement transmet les informations appropriées à votre fonction gestionnaire d'événement. Seul l'événement **OnTermBaseRegister** ne transmet aucune donnée. Tous les autres transmettent au moins l'ID du terminal sur lequel s'est produit l'événement. **OnTermData** transmet également les données tapées ou scannées du terminal.

La valeur d'un ID de terminal est toujours comprise entre 0 et 15. La valeur transmise 99 indique une erreur.

Une fois que vous avez les structures de base de gestionnaire d'événement, vous pouvez ajouter aux événements toutes les fonctionnalités que vous souhaitez.

La méthode **OpenDevice** doit être appelée avant le lancement de n'importe quel événement.

OnTermBaseRegister
Evénement : Une station de base connectée est parvenue à se mettre en marche et à communiquer avec l'ordinateur hôte via la connexion de série.

OnTermSignIn6
Données transmises : terminal
Evénement : Un terminal à six lignes vient de se connecter. L'ID de terminal est transmis au terminal.

OnTermSignIn4
Données transmises : terminal
Evénement : Un terminal à quatre lignes vient de se connecter. L'ID de terminal est transmis au terminal.

OnTermSignOut
Données transmises : terminal
Evénement : Un terminal vient de se déconnecter. L'ID de terminal est transmis au terminal.

OnTermData
Données transmises : terminal, data
Evénement : Un terminal a renvoyé des données en réponse à un appel de méthode d'entrée.

OnTermNotSignedIn
Données transmises : terminal
Evénement : Une commande a été transmise à un terminal non connecté.

OnTermSequenceError
Données transmises : terminal
Evénement : Violation du protocole un-pour-un (une invite de l'hôte contre une réponse du terminal). L'hôte ne peut pas envoyer une deuxième commande d'entrée tant qu'il n'aura pas reçu de réponse à la première commande. Si une station de base

reçoit 5 erreurs de séquence de suite, une erreur de logique d'hôte est générée et la base s'éteint.

Bien que le contrôle ActiveX de PromptCom intercepte et empêche la plupart des erreurs de logique, celles-ci peuvent se produire. Nous vous conseillons d'implémenter ce gestionnaire d'événement.

OnTermIllegalCommand
Données transmises : terminal
Evénement : Une commande illégale a été transmise à un terminal.

Le contrôle ActiveX de PromptCom sert à empêcher les commandes illégales, mais le logiciel n'est pas infaillible et nous ne pouvons pas prévoir toutes les situations rencontrées par nos clients !

OnTermUpArrow
Données transmises : terminal
Evénement : La touche flèche vers le haut d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que *WDterm* puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermDownArrow
Données transmises : terminal
Evénement : La touche flèche vers le bas d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que *WDterm* puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermLeftArrow
Données transmises : terminal
Evénement : La touche flèche vers la gauche d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que *WDterm* puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermRightArrow
Données transmises : terminal
Evénement : La touche flèche vers la droite d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que *WDterm* puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermBeginKey
Données transmises : terminal
Evénement : La touche BEGIN d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que *WDterm* puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermEndKey
Données transmises : terminal
Evénement : La touche END d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que *WDterm* puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermSearchKey
Données transmises : terminal
Evénement : La touche SEARCH d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que *WDterm* puisse répondre de nouveau à la pression d'une touche du terminal.

Contrôles ActiveX TCP/IP de PromptNET

Le contrôle ActiveX de PromptNET est un composant drop in de communication qui permet aux programmeurs d'ajouter facilement une capacité de transmission d'invites et réception de données en provenance de leur terminal radio via la station de base et une connexion réseau TCP/IP.

PromptNET doit disposer d'un ordinateur "client" sur le réseau TCP/IP (auquel 4 stations de base peuvent être connectées) ainsi que d'un ordinateur "serveur" visible sur le réseau par le client.

L'ordinateur client lance le programme des utilitaires client PromptNET en arrière plan. L'ordinateur serveur lance votre application qui utilise le composant ActiveX PromptNET pour communiquer avec le client.

Le composant ActiveX est compatible avec Visual Basic, Visual C++, Delphi et la plupart des plate-formes de développement en 32-bits. Le programme client fonctionne sous l'environnement Windows 98 ou ultérieur. Pour obtenir des instructions d'installation, consultez le fichier d'aide.

Programmation

Configuration réseau

- Les paramètres réseau client et serveur doivent prendre en charge les communications TCP/IP.

- Les ordinateurs client et serveur doivent impérativement communiquer entre eux sur votre réseau. Chacun doit posséder une adresse IP dans le même sous-réseau. Le serveur requiert une adresse IP statique alors que le client peut avoir une adresse statique ou attribuée par un serveur DHCP ou équivalent. Pour configurer vos paramètres d'adresse IP, reportez-vous à l'utilitaire Réseau dans le Panneau de configuration de Windows.

- PromptNET utilise les ports 54123 (serveur) et 54124 (client).

- Vous pouvez relier le serveur et le client à l'aide d'une connexion Internet à distance ou de type DSL, à condition que le serveur possède une adresse IP statique et que votre routeur utilise les ports ci-dessus.

- Si vous n'êtes pas sûr de pouvoir paramétrer correctement votre configuration IP, contactez votre administrateur réseau.

Utilitaire client

Assurez-vous que l'utilitaire client est installé correctement sur l'ordinateur client et qu'il communique avec au moins une station de base. Testez le client en allumant la station de base. Le message "**Base SignOn**" doit s'afficher à l'écran.

Communications serveur

Lancez l'utilitaire de test du serveur sur l'ordinateur serveur. A partir de l'ordinateur client, définissez l'adresse IP de l'ordinateur serveur ainsi qu'un "Base Name" (nom de base) unique pour l'utilitaire client. Essayez ensuite de vous connecter à l'utilitaire de test du serveur. Si l'utilitaire client se connecte, cela signifie que votre configuration est valide. A partir de l'ordinateur serveur, fermez l'utilitaire de test du serveur. Vous pouvez maintenant travailler sur l'application PromptNET de votre serveur.

Pour la communication client-serveur, l'utilitaire client doit être exécuté sur le PC auquel sont connectées les stations de base série.

Avant de lancer des appels de méthode WDIPterm dans votre application, assurez-vous de définir la propriété **ServerOn** à "True".

Test de qualité de la communication
Implémentez un gestionnaire d'événement pour *OnTermBaseRegister* qui provoque un signal sonore ou l'affichage d'un message lorsqu'il est appelé. Si la communication fonctionne normalement entre l'hôte et la station de base, votre gestionnaire d'événement s'exécutera dès l'activation de votre programme et la mise en marche d'une station de base connectée.

Stations de base multiples
Dans les installations où plusieurs stations de base sont connectées à un PC client unique, utilisez simplement les quatre "canaux" du programme d'utilitaire client.

Suivi de terminal
Comme vous disposez d'une seule série de gestionnaires d'événements, vous devez élaborer un plan qui vous permettra d'assurer le suivi de chaque terminal (jusqu'à 16 terminaux par station de base et jusqu'à 4 stations de base par client) dans sa séquence de transactions. Vous pouvez par exemple utiliser une variable "state" ("état", stockée dans un tableau) pour chaque terminal. Testez cette variable pour déterminer l'invite suivante de n'importe quel terminal. Pour plus d'informations, reportez-vous aux échantillons.

L'"état de connexion" de chaque terminal doit être suivi avec la plus grande attention. Chaque événement **SignOut** (déconnexion) doit être associé à un événement **SignIn** (connexion) et aucun terminal ne doit pas être autorisé à se connecter deux fois sans événement **SignOut** préalable. Lorsque plusieurs événements **SignIn** proviennent d'un terminal sans événement **SignOut**, cela indique l'un des problèmes suivants :

1) Un terminal hors de portée est mis en marche avant un retour dans la zone de portée

2) Deux terminaux ou plus utilisent le même identificateur (conflit d'ID).

Concepts

Les composants drop-in s'ajoutent à vos outils de programmation. Seules les technologies ActiveX sont compatibles avec la plupart des environnements de développement. Lorsque vous utilisez des composants drop-in dans votre programme, vous suivez le paradigme standard de programmation orientée-objet qui implémente la fonctionnalité du composant drop-in à l'aide de propriétés, de méthodes et d'événements.

Les **propriétés** correspondent aux diverses variables de configuration utilisées par le composant drop-in. Le paramètre ServerOn en est un exemple.

Les **méthodes** sont des appels de fonction qui fournissent les commandes et les fonctionnalités d'accès du composant drop-in. Vous pouvez par exemple envoyer une commande Input (entrée) au terminal.

Les **événements** sont des définitions de fonction placées dans le code source de votre application. Dans ce dernier, ces définitions sont appelées Event Handlers (gestionnaires d'événements). La structure de base du code source du gestionnaire d'événements est générée automatiquement. Le code du gestionnaire d'événement est appelé (ou lancé) par le composant drop-in lorsqu'un événement spécifique se produit ; par exemple, un terminal renvoie des données et l'événement OnTermData est lancé.

Les détails relatifs à l'accès aux propriétés, aux méthodes et aux événements varient selon les plate-formes de développement. Le mode de fonctionnement détaillé des plate-formes les plus connues est illustré dans les échantillons du CD-Rom des Utilitaires ou téléchargeables à partir de notre site :

http://www.barcodehq.com/wdterminal.exe

Propriétés

Les propriétés sont les diverses variables de configuration utilisées par le contrôle WDIPterm. Elles peuvent être assignées directement dans votre application (par exemple, " WDIPterm.ServerOn = true ") et définies dans l'explorateur d'objets de votre environnement de développement.

Remarque : Pour le contrôle WDIPterm, votre environnement de développement peut contenir des propriétés supplémentaires et absentes de cette liste. Ceci est normal et vous pouvez ignorer ces propriétés.

ServerOn
Valeurs possibles : True, False (vrai, faux)
Fonction : Définir à True pour lancer le serveur. Définir à False pour éteindre le serveur. Si votre programme n'est pas en cours d'exécution, désactivez cette propriété. Si vous la définissez à True lors de la conception, vous risquez de rencontrer des problèmes.

Quiet
Valeurs possibles : True, False
Fonction : Si le paramètre Quiet est défini à True, tous les messages d'état et d'erreur générés par WDIPterm sont supprimés.

LogFile
Valeurs possibles : vide ou un nom de fichier valide
Fonction : Si le journal n'existe pas, il est créé. S'il existe, il sera ajouté. Si la valeur LogFile est vide, aucun fichier journal n'est mis à jour.

ClientList
Valeurs possibles : Read Only (lecture seule)
Fonction : Renvoie une chaîne mise en forme qui répertorie tous les noms de base client et les numéros IP associés. Le format est le suivant : "basename CR/LF ip address CR/LF basename…" ("nom de la base, retour chariot, adresse ip, retour chariot, nom de la base...").

Méthodes

Les méthodes sont des commandes que vous fournissez au contrôle WDIPterm. Toutes les commandes "Inputxxx" indiquent au terminal d'attendre que l'opérateur entre des données.

Remarque : Pour le contrôle WDterm, votre environnement de développement peut indiquer d'autres méthodes qui ne figurent pas dans cette liste. Ceci est normal et vous pouvez ignorer ces propriétés.

InputAny
Paramètres : basename, channel, terminal, line, position, prompt, shifted, timestamped
Fonction : Cette méthode instruit le terminal connecté à la base client sur le canal d'afficher l'invite au bon endroit, puis d'attendre les données saisies via le clavier ou le scanner du terminal. Si le paramètre "shifted" (verrouillage majuscules) est défini à True, le terminal démarre en mode shifted. Le paramètre "timestamped" (heure) ajoute un préfixe (au format hhmmss) aux données retournées.

InputKeyBd

Paramètres : basename, channel, terminal, line, position, prompt, shifted, timestamped

Fonction : Cette méthode instruit le terminal connecté à la base client sur le canal d'afficher l'invite au bon endroit, puis d'attendre les données saisies via le clavier du terminal. Si le paramètre "shifted" (verrouillage majuscules) est défini à True, le terminal démarre en mode shifted. Le paramètre "timestamped" (heure) ajoute un préfixe (au format hhmmss) aux données retournées.

InputScanner

Paramètres : basename, channel, terminal, line, position, prompt, allowbreakout, timestamped

Fonction : Cette méthode instruit le terminal connecté à la base client sur le canal d'afficher l'invite au bon endroit, puis d'attendre les données saisies via le scanner du terminal. Définir la valeur allowbreakout à True permet à l'utilisateur de désactiver le mode scanner seul en appuyant sur la touche End du terminal. Un termID+CR sera envoyé à l'hôte.

InputYesNo

Paramètres : basename, channel, terminal, line, position, prompt

Fonction : Cette méthode indique au terminal connecté à la base client sur le canal d'afficher l'invite au bon endroit, puis d'attendre la saisie de oui (touche Enter ou C) ou de non (Touche 0 ou B) à l'aide clavier du terminal.
Remarque : les touches C et B sont utilisées pour faciliter la saisie clavier lorsque la lecture avec le laser intégré est activée.

InputPassword

Paramètres : basename, channel, terminal, line, position, prompt, shifted

Fonction : Cette méthode indique au terminal connecté à la base client sur le canal d'afficher l'invite au bon endroit, puis d'attendre les données saisies via le clavier du terminal. Les données saisies ne s'affichent pas sur le terminal.

InputSerial

Paramètres : basename, channel, terminal, line, position, prompt

Fonction : Cette méthode indique au terminal connecté à la base client sur le canal d'afficher l'invite au bon endroit, puis d'attendre la réception de données via le port série du terminal. L'attente en entrée série peut être évitée en appuyant sur la touche Enter du terminal. Celui-ci enverra une chaîne de données vide à l'hôte (lance le gestionnaire d'événement OnTermData).

OutputSerial

Paramètres : basename, channel, terminal, data

Fonction : Cette méthode indique au terminal connecté à la base client sur le canal d'envoyer des données au port série du terminal. Les données ne doivent pas comporter plus de 248 caractères pour chaque appel vers OutputSerial. Si vous envoyez des données à une imprimante connectée au terminal, assurez-vous de définir le paramètre Protocole du terminal radio à XON/XOFF. Pour plus d'informations, consultez le manuel du terminal radio.

SendDisplay

Paramètres : basename, channel, terminal, line, position, prompt

Fonction : Cette méthode indique au terminal connecté à la base client sur le canal d'afficher l'invite au bon endroit. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

ClearScreen
Paramètres : basename, channel, terminal
Fonction : Cette méthode indique au terminal connecté à la base client sur le canal d'afficher l'invite au bon endroit. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

ClearLine
Paramètres : basename, channel, terminal, line
Fonction : Cette méthode indique au terminal connecté à la base client sur le canal d'afficher l'invite au bon endroit. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

SendDate
Paramètres : basename, channel, terminal, line
Fonction : Cette méthode indique au terminal connecté à la base client sur le canal d'afficher la date et l'heure au bon endroit. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

Signal sonore
Paramètres : basename, channel, terminal, count
Fonction : Cette méthode indique au terminal connecté à la base client sur le canal de déclencher un signal sonore. Le décompte peut avoir une valeur comprise entre 1 et 9. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

PlayVoice
Paramètres : basename, channel, terminal, msgnum
Fonction : Cette méthode indique au terminal connecté à la base client sur le canal de diffuser le message vocal "msgnum". Msgnum peut avoir une valeur comprise entre 1 et 99 et doit être suivi d'un appel de méthode Input (entrée) pour être activé.

ReInit
Paramètres : basename, channel, terminal
Fonction : Cette méthode indique au terminal connecté à la base client sur le canal de se réinitialiser. Elle doit être suivie d'un appel de méthode Input (entrée) pour être activée.

Remarque : Le message "base réinitialisée" s'affichera sur les terminaux des stations de base qui utilisent des versions d'EEPROM antérieures à 9079. Seul le terminal a été réinitialisé. Les dernières stations de base affichent le message "Buffer Reinitialized..." pour indiquer la rénitialisation d'un seul terminal.

ReInitAll
Paramètres : basename, channel
Fonction : Cette méthode instruit tous les terminaux connectés à la base client sur le canal de se réinitialiser.

TestClient
Paramètres : aucun
Fonction : Cette méthode indique au serveur d'exécuter la commande Ping pour tous les clients connectés. Les résultats sont enregistrés dans le journal.

Evénements

Les événements WDIPterm requièrent des conditions spécifiques pour se déclencher. Lorsqu'un événement est lancé, une fonction de gestionnaire d'événement de votre application est appelée.

Bien que la façon dont les événements se produisent varie d'un environnement de programmation à un autre, les structures de base des codes sources des

divers gestionnaires d'événement sont générés automatiquement et insérés dans votre code source. Pour plus d'informations, reportez-vous aux exemples.

Chaque événement transmet les informations appropriées à votre fonction gestionnaire d'événement. OnTermData transmet également les données tapées ou scannées du terminal.

La valeur d'un ID terminal est toujours comprise entre 0 et 15. La valeur 99 correspond à l'emplacement réservé pour la connexion.

Une fois que vous avez les structures de base de gestionnaire d'événement, vous pouvez ajouter aux événements toutes les fonctionnalités que vous souhaitez.

Vous devez définir la propriété ServerOn à True avant le lancement de n'importe quel événement.

OnTermBaseRegister
Données transmises : basename, channel
Evénement : Une station de base connectée à la base client sur le canal est parvenue à se mettre en marche et à communiquer avec l'ordinateur hôte via la connexion de série.

OnTermSignIn6
Données transmises : basename, channel, terminal
Evénement : Un terminal à six lignes s'est connecté à la base client sur le canal. L'ID de terminal est transmis au terminal.

OnTermSignIn4
Données transmises : basename, channel, terminal
Evénement : Un terminal à quatre lignes s'est connecté à la base client sur le canal. L'ID de terminal est transmis au terminal.

OnTermSignOut
Données transmises : basename, channel, terminal
Evénement : Un terminal s'est déconnecté du canal à la base client sur le canal. L'ID de terminal est transmis au terminal.

OnTermData
Données transmises : basename, channel, terminal, data
Evénement : Un terminal connecté à la base client sur le canal a envoyé des données en réponse à un appel de méthode d'entrée.

OnTermNotSignedIn
Données transmises : basename, channel, terminal
Evénement : Une commande a été transmise à un terminal non connecté.

OnTermSequenceError
Données transmises : basename, channel, terminal
Evénement : Violation du protocole un-pour-un (une invite de l'hôte contre une réponse du terminal). L'hôte ne peut pas envoyer une deuxième commande d'entrée tant qu'il n'aura pas reçu de réponse à la première commande. Si une station de base reçoit 5 erreurs de séquence de suite, une erreur de logique d'hôte est générée et la base s'éteint.

Bien que le contrôle ActiveX de PromptCom intercepte et empêche la plupart des erreurs de logique, celles-ci peuvent se produire. Nous vous conseillons d'implémenter ce gestionnaire d'événement.

OnTermIllegalCommand
Données transmises : basename, channel, terminal
Evénement : Une commande illégale a été transmise à un terminal.

Le contrôle ActiveX de PromptCom sert à empêcher les commandes illégales, mais le logiciel n'est pas infaillible et nous ne pouvons pas prévoir toutes les situations rencontrées par nos clients !

OnTermUpArrow
Données transmises : basename, channel, terminal
Evénement : La touche flèche vers le haut d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que WDIPterm puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermDownArrow
Données transmises : basename, channel, terminal
Evénement : La touche flèche vers le bas d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que WDIPterm puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermLeftArrow
Données transmises : basename, channel, terminal
Evénement : La touche flèche vers la gauche d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que WDIPterm puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermRightArrow
Données transmises : basename, channel, terminal
Evénement : La touche flèche vers la droite d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que WDIPterm puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermEndKey
Données transmises : basename, channel, terminal
Evénement : La touche END d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que WDIPterm puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermBeginKey
Données transmises : basename, channel, terminal
Evénement : La touche BEGIN d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que WDIPterm puisse répondre de nouveau à la pression d'une touche du terminal.

OnTermSearchKey
Données transmises : basename, channel, terminal
Evénement : La touche SEARCH d'un terminal a été pressée. Vous devez fournir un autre appel de méthode d'entrée si vous souhaitez que WDIPterm puisse répondre de nouveau à la pression d'une touche du terminal.

Bibliothèque DLL PromptCOM pour Windows

Le programme de la bibliothèque DLL PromptCOM s'exécute avec les environnements de programmation Windows qui ne peuvent pas utiliser les composants ActiveX. Dans la mesure du possible, nous vous conseillons d'utiliser les composants ActiveX.

Le programme est disponible sur le CD-ROM des utilitaires du terminal radio ou à partir de notre site : http://www.barcodehq.com/winterm.exe. Pour l'installer, double-cliquez sur INSTALL.EXE dans l'explorateur Windows. La bibliothèque DLL PromptCOM pour Windows existe en deux versions, en 16 et en 32 bits, qui permettent aux programmeurs d'ajouter facilement les

fonctionnalités de transmission d'invites et de réception de données en provenance de leur terminal radio, via la station de base ou une communication série directe.

L'interface de programmation d'applications (API) pour PromptCOM offre les fonctions suivantes :

InitComDLL Initialise la fonction InitComDLL du système PromptCOM.

CloseComDLL Eteint le système PromptCOM et libère des ressources sans fermer l'application parente.

Setup La fonction Setup sert à la configuration du port de communication.

SendCommand Cette fonction envoie une commande au terminal doté de l'ID correspondant.

GetCommData Cette fonction retourne les données saisies par l'unité distante en réponse à l'invite.

DataAvailable A l'aide de cette fonction vous pouvez vérifier la présence de données à traiter avant d'appeler GetCommData.

Vous trouverez des échantillons de programmes sur la disquette pour Visual Basic, Access et Delphi. Il existe également un exemple de code Visual Basic qui ne requiert pas la bibliothèque DLL. Consultez l'aide ainsi que le fichier LISEZMOI pour prendre connaissance des toutes dernières modifications.

Commandes de programmation de bas niveau

Les commandes de programmation de bas niveau sont fournies pour les configurations qui n'utilisent pas les contrôles ActiveX (ex. : programmation sous UNIX, etc.).

Programmation de l'hôte au terminal

L'ID du terminal radio correspond au premier octet et possède toujours une longueur d'un seul caractère. Il existe 16 valeurs différentes (chiffres de 1 à 9 et lettres de A à F).

La section Commande(s) du message commence toujours au deuxième octet et peut être composée d'une ou plusieurs commandes, y compris des commandes du type affichage de données ou diffusion de messages vocaux.

En fin de message, le dernier octet correspond toujours au caractère ASCII 4 (EOT).

Vous trouverez ci-dessous une liste des commandes valides et quelques exemples :

Fonction de commande	Caractères de commande
*@	Réinitialise tous les terminaux
3@	Réinitialise le terminal n°3
1@Bn	Déclenche un signal sonore n fois (1 à 9) sur le terminal n°1
2@C0*	Efface les données affichées (sur 4 ou 6 lignes) du terminal n°2
0@C1	Efface la 1ère ligne du terminal n°0
1@C2	Efface la 2ème ligne du terminal n°1

2@C3 ..... Efface la 3ème ligne du terminal n°2
0@C4 ..... Efface la 4ème ligne du terminal n°0
3@C5* ..... Efface la 5ème ligne du terminal n°3 s'il s'agit d'un affichage à 6 lignes ; efface tout s'il s'agit d'un affichage à 4 lignes.
1@C6* ..... Efface la 6ème ligne d'un terminal à 6 lignes. N'a aucun effet sur un terminal à 4 lignes.
1@Dn ..... Affiche la date et l'heure sur les lignes 1 à 4 du terminal n°1 au format américain (mm/jj/aa, hh:mm:ss) ou au format européen (jj/mm/aa, hh:mm:ss).
1@Vnn ..... Diffuse un message vocal n°nn (1 à 99) sur le terminal n°1.
1@Sdataxxxx ..... Envoie les données "xxxxxxx" au port série du terminal n°1, 255 caractères maximum.

Le format ci-dessous est suivi d'une séquence typique de commande d'invite :

0@n,m,o,data

n est le numéro de la ligne (1 à 4) à laquelle l'invite doit s'afficher.
m est la position de caractère (1 à 20) à laquelle l'invite doit s'afficher.
o est le caractère qui détermine si l'invite est destinée à l'affichage seul (0) ou si elle attend l'envoi de données (1). Pour obtenir les caractères valides de cette position, consultez le tableau ci-dessous.
data représente les données que vous souhaitez afficher.

Par exemple, passer la commande **@1,1,1, Entrer quantité** provoquerait l'affichage du message "Entrer quantité" à la position 1 de la ligne 1. L'opérateur devrait ensuite entrer les données correspondantes.

Vous trouverez ci-dessous des entrées valides pour les caractères de troisième position :

0 .....Aucun d'envoi de données pour cette commande, affichage seul.
1 .....Entrée de données via le clavier ou le scanner requise.
2 .....Entrée de données via le clavier seulement, verrouillage majuscules désactivé.
3 .....Entrée de données via le clavier seulement, verrouillage majuscules activé.
4 .....Entrée de données via le scanner seulement.
5 .....Accepter uniquement les réponses OUI, touche Enter ou NO, touche 0 du clavier (le terminal envoie 1 pour OUI et 0 pour NON).
A .....Identique à 1 mais avec le préfixe time stamped (hhmmss).
B .....Identique à 2 mais avec le préfixe time stamped (hhmmss).
C .....Identique à 3 mais avec le préfixe time stamped.
D .....Identique à 4 mais avec le préfixe time stamped.
E .....Identique à 4 mais avec la possibilité d'appuyer sur la touche END pour sortir du mode entrée scanner seul. ID et CR du terminal envoyés à l'hôte.
S .....Entrée clavier verrouillage majuscules activé ou scanner.
p .....Entrée clavier verrouillage majuscules désactivé en mode sans affichage (pour mots de passe).
P .....Entrée clavier verrouillage majuscules activé en mode sans affichage (pour mots de passe).
R .....Entrée de données requise par le port série RS-232. L'attente en entrée série peut être évitée en appuyant sur la touche ENTER qui renverra une chaîne de données NULL (vide) à l'ordinateur hôte. Ce caractère est utilisé pour les scanners série PDF 417 et les saisies sur bande magnétique des imprimantes Caméo. Un terminal

de point de vente devient alors possible. Scannez la carte de crédit et imprimez le reçu à partir du terminal radio.

Vous trouverez ci-dessous quelques règles et conseils pour créer des commandes :

- Les messages peuvent regrouper plusieurs commandes (messages vocaux, initialisation, suppression de lignes, requête d'entrée de données) pouvant comporter jusqu'à 247 caractères. Un message ne peut cependant pas contenir la commande @S en combinaison avec d'autres commandes. En outre, un message ne doit pas contenir plus d'une requête d'entrée de données (le troisième caractère de la commande étant 1). Par exemple...

  @1,1,1,ITEM@2,1,1,QTY

  ... est composée de deux commandes d'entrée de données. Si ce message était envoyé au terminal radio, la première invite d'entrée de données (@1,1,1,ITEM) serait exécutée mais toutes les commandes suivantes seraient ignorées sans avertissement. Aucun affichage ou indication ne signalerait une commande illégale.

- La commande @S (en entrée série) ne peut pas être combinée à d'autres commandes. Lorsqu'une commande @S a été exécutée avec succès, la station de base renvoie à l'hôte l'ID du terminal radio suivi d'un CR (ASCII 13). Cette commande est limitée à 247 caractères. Si vous envoyez une commande de plus de 247 caractères vous obtiendrez un message de commande illégale (ID ? CR). Si vous devez envoyer une commande de 300 caractères, envoyez une première partie, attendez une réponse positive (ID CR), puis envoyez la partie restante.

  Si vous utilisez la commande @S avec une imprimante, assurez-vous de définir le paramètre Protocole du terminal radio à XON/XOFF. Cela permettra au terminal radio de gérer les restrictions de longueur de caractère de la mémoire tampon de votre imprimante. Si vous utilisez l'imprimante O'Neil MicroFlash, vous devez envoyer un caractère NULL en tête des données valides pour éveiller l'imprimante.

  Si vous utilisez une imprimante Caméo ou Encore, vous devez posséder un terminal capable d'afficher Z sur la ligne de la fréquence sur l'écran d'ouverture. Si le Z n'est pas disponible, l'imprimante ne s'éveillera pas. Les imprimantes Encore et Caméo s'éveillent lorsque leur ligne DSR est stimulée. Si le problème est signalé lors de la commande, une modification matérielle sera apportée gratuitement au terminal radio.

- Toutes les déclarations doivent se terminer par une commande d'invite d'entrée de données, que la déclaration soit composée d'une commande unique ou de plusieurs commandes combinées. Les déclarations illégales seront ignorées mais seront affichées sur le terminal radio de destination exactement comme elles avaient été rédigées. Si aucun ID de terminal n'a été inscrit dans la déclaration, il essaiera d'afficher la déclaration non valide de l'ID 0. Sur pression de la touche ENTER lors de l'affichage de la déclaration non valide, le terminal envoie le caractère "?" à la station de base. En retour, la station de base envoie à l'ordinateur hôte le message n?CR (n correspond à l'ID de terminal et CR au retour chariot). Les versions du terminal radio antérieures à la version 9.059 ne géraient pas les déclarations illégales de la même manière. Pour une compatibilité descendante, voir le paramètre de configuration du Protocole (modifiez-le en E).

- Avant la version 9075, le terminal radio n'était disponible qu'en affichage sur 4 lignes. Certaines commandes fonctionnent différemment

selon le nombre de lignes d'affichage, en particulier le caractère SIGN ON. Pour plus d'informations, consultez le manuel du terminal radio.

Vous trouverez ci-dessous quelques exemples de déclarations de commandes :

@2,1,1,ENTER ITEM NO	Affiche le message ENTER ITEM NO (ENTRER NO ARTICLE) à la ligne n°2, position 1 et attend l'entrée des données.
@V23@1,2,1,WRONG ITEM	Diffuse le message vocal 23, affiche le message WRONG ITEM (ARTICLE INCORRECT) à la ligne n°1, position 2. Requiert ensuite une attente d'entrée de données.
@C1@1,7,0,PICKING	Efface la ligne n°1. Affiche le message PICKING (SELECTION) à la ligne n°1, position 7. Cette commande est illégale. Pour être validée, elle doit se terminer par une requête d'entrée de données telle que : **@C1@1,7,0,PICKING@2,7,1,ITEM**.
@1,1,1,ITEM@2,1,1,QTY	Comme une commande peut être une requête d'entrée de données ("invite"), cette déclaration est illégale et sera par conséquent ignorée en tant que commande.

Formats des transmissions de la station de base à l'hôte

Le format de base d'un message transmis de la station de base à l'hôte est simple :

Position des octets	Fonction	Valeurs possibles
1	ID du terminal radio	0-F
2+	Données transmises	**
Dernier octet	Marque la fin du message	CR (ASCII 13)

En général, la station de base envoie une réponse à la question de l'hôte. Par exemple, si une base envoie le message d'hôte ci-dessous au terminal n°2...

2@1,1,1,ITEM NUMBER + EOT

... le message ITEM NUMBER (NUMERO ARTICLE) s'affichera sur l'écran du terminal radio à la ligne n°1, position 1. L'opérateur entrera alors un numéro d'article à l'aide du scanner ou du clavier. Le terminal radio transmet les données entrées (par exemple 123) à la station de base qui à son tour transmet le message suivant à l'hôte :

2123+CR

Où **2** correspond à l'ID du terminal
123 correspond aux données
CR marque la fin du message

En plus des données, la station de base peut envoyer d'autres messages à l'hôte :

Par exemple, si le terminal n°2 reçoit une commande illégale, la station de base transmet le message suivant à l'hôte :

2?CR

Serial Reply (Réponse série)
Lorsqu'une commande série a été exécutée correctement, la station de base envoie à l'hôte l'ID de terminal suivi d'un Retour Chariot (CR). Les commandes série sont généralement utilisées avec les imprimantes série connectées. Les commandes série envoyées à la station de base et au terminal ne peuvent pas être associées à d'autres commandes dans le même message. Notez qu'il est impossible d'envoyer des commandes dépassant 247 caractères, ID, @S et EOT inclus.

Position des octets	Fonction	Valeurs possibles
1	ID du terminal radio	0-F
2+	SIGN ON	SI (ASCII 15) si le terminal dispose d'un affichage 4 lignes ou d'un affichage 6 lignes configuré comme un 4 lignes dans la version du microprogramme 9075 ou supérieure. SYN (ASCII 22) si le terminal 6 lignes est configuré comme tel dans la version du microprogramme 9075 ou supérieure.
Dernier octet	Marque la fin du message	CR (ASCII 13)

Les versions de terminaux antérieures à 9075 se connectaient à l'aide du caractère ASCII 15.

A partir de la version 9075, le caractère SIGN ON a été ajouté pour différencier les terminaux à 4 lignes et de ceux à 6 lignes. Dans la version actuelle, le caractère SIGN ON d'un terminal à 6 lignes configuré comme tel envoie le caractère ASCII 22. Le caractère SIGN ON d'un terminal à 6 lignes configuré pour un affichage à 4 lignes transmet le caractère ASCII 15. Tous les terminaux disposant d'un affichage à 4 lignes, quelle que soit leur version, transmettent le caractère de ASCII 15 (SIGN ON).

Si la station de base reçoit 5 erreurs de séquence de suite, elle transmet le message ci-dessous au terminal avant de s'éteindre :

Base Shut Down
Due to Host Logic
Error

Avant de reprendre, recherchez les erreurs dans votre programme. Relancez ensuite la station de base et reconnecter le terminal pour continuer.

Si le terminal est connecté (SIGN ON) au système et que la station de base est réinitialisée, le message suivant apparaît sur le terminal :

Base Reinitialized X
Cycle Power on RF
Terminal and Sign-on
again to Restart_

X peut correspondre à "P" (l'initialisation de la base est liée à l'alimentation) ou à "H" (l'initialisation de la base est liée à l'hôte). Si

votre station de base est équipée d’une mémoire EPROM antérieure à RFX9079D, seul le "P" s’affichera.

Transmission des caractères ASCII à l’aide du clavier du terminal