Hub One – Mobility

TouchKbd 3.1

Spécifications fonctionnelles

David DESVAUX

Historique du document

Date	Auteur(s)	Modification
27/03/2013	David DESVAUX	Multi-clavier / couleur de fond.
29/03/2013	David DESVAUX	Nom / taille de police.
03/04/2013	David DESVAUX	Couleur fond & police par bouton.
10/04/2013	David DESVAUX	Suppression de la méthode de configuration du style temporaire et ajout des styles.
11/04/2013	David DESVAUX	Modification de la définition des styles.
12/04/2013	David DESVAUX	Suppression de la partie concernant la modification du comportement des valeurs rowspan et colspan.
16/09/2013	David DESVAUX	Ajout des commandes et des dégradés.
21/11/2013	David DESVAUX	Messages SHOW/HIDE/TOGGLE et clé HWND
14/04/2015	David DESVAUX	Nouveaux messages pour manip depuis scripting

Sommaire

1 Conventions

Ce document a pour fonction de lister des exigences et des spécifications. Pour pouvoir clairement identifier chaque exigence/spécification, ce document met en place une nomenclature. Cette dernière attribue un identifiant unique à chaque exigence/spécification. Les identifiants en question possèdent une syntaxe qui permet de situer clairement une exigence/spécification. Ces identifiants seront utilisés lors du développement et des tests. En voici la syntaxe :

Domaine.*Sous-domaine*.*Numéro*

où:

- *Domaine* indique le domaine de l'exigence parmi :
  - **Fonc** (Fonctionnel),
  - **Env** (Environnement).
- *Sous-domaine* représente le sous-domaine d'exigence.
  Pour le domaine Fonc (Fonctionnel), les sous-domaines sont :
  - **Para** (Paramétrage)
  - **Lect** (Lecture)
  - **Cont** (Contrôle)

  Pour le domaine Env (Environnement), les sous-domaines sont :
  - **Logi** (Logiciel)
  - **Mate** (Materiel)

- *Numéro* représente le numéro unique de l'exigence au sein du domaine.

Exemple :

Fonc.Para.001 est l'identifiant de l'exigence fonctionnelle n°1 du sous-domaine Paramétrage.

2 Existant

2.1 TouchKbd 2.0

Le programme permet d'afficher un clavier virtuel à l'écran. L'appui sur l'une des touches de ce clavier simule l'appui d'une touche et permet donc d'utiliser un terminal sans avoir recours à un clavier physique.

Ce clavier peut disposer de 1 à N « layouts » différentes (un ensemble de touches disposées d'une manière explicitement définie) entre lesquelles il est possible de naviguer en effectuant un déplacement horizontal sur l'écran.

Ces layouts, ainsi que les paramètres généraux du clavier sont définis dans le fichier de configuration du « telnet » au format « .cfg » qui est structuré comme un fichier au format « .ini ». Lors du premier lancement de TouchKbd depuis le telnet, ce fichier est lu, et les réglages sont stockés dans le registre du système d'exploitation.

2.2 Fichier de configuration

La configuration de TouchKbd est définie à l'intérieur du fichier de configuration de telnet.

Pour chaque layout il y a un « bloc » associé. La première layout, obligatoire, est définie dans le bloc « [TOUCK_KBD] », et les suivantes, facultatives, le sont dans les blocks « [TOUCH_KBD_1] », « [TOUCH_KBD_2] », etc.

La configuration générale du clavier (taille, position, etc) est définie dans le bloc de la première layout comme suit :

```
[TOUCH_KBD]                 Paramètres généraux du clavier et première layout
ShowTouchKbd=Yes            Si l'on affiche le clavier
PosX=0                      Position sur l'axe horizontal, en pixels.
PosY=160                    Position sur l'axe vertical, en pixels.
SizeX=240                   Largeur, en pixels.
SizeY=160                   Hauteur, en pixels.
Layouts=3                   Nombre des « layouts » différentes.
KeyCount=24                 Nombre de touches définies
Key01=1,1,N,0x70,F1         Définition d'une première touche
Key02=1,2,N,0x71,F2         Définition d'une seconde touche
Key03=1,3,N,0x72,F3         Etc…

[TOUCH_KBD_1]               Deuxième layout
KeyCount=24                 Nombre de touches définies
Key01=1,1,N,0x70,F1         Définition d'une première touche
Key02=1,2,N,0x71,F2         Définition d'une seconde touche
Key03=1,3,N,0x72,F3         Etc…

[TOUCH_KBD_2]               Troisème layout
KeyCount=24                 Nombre de touches définies
Key01=1,1,N,0x70,F1         Définition d'une première touche
Key02=1,2,N,0x71,F2         Définition d'une seconde touche
Key03=1,3,N,0x72,F3         Etc…
Etc…
```

2.2.1 Syntaxe pour la définition d'une touche.

Conventions pour la syntaxe des touches :

- « [contenu] » indique 0 ou 1 fois le contenu. Par exemple « A[B]C » peut indiquer « AC » ou ABC ».
- « {contenu} » indique 0 ou N fois le contenu. Par exemple « A{B}C » peut indiquer « AC » ou « ABC » ou encore « ABBB…BBBBBC ».
- « | » indique « ou ». Par example : « A|B » peut indiquer « A » ou « B »

- Syntaxe minimale :

```
ligne,colonne,état,,texte
Example :
1,1,N,,texte
```

Le « key-code » (voir ci après) est désormais facultatif, d'où les deux virgules consécutives.

- Syntaxe avec attribution d'une style (de numéro N, précédé de « ¤ ») :

```
ligne,colonne,état,,texte¤N
Example :
1,1,N,,texte¤1
```

- Syntaxe définissant une touche de taille différente de 1x1 :

```
ligne,colonne,état,,texte,lignes,colonnes
Example :
1,1,N,,texte,2,2
```

- Syntaxe définissant une touche simulant un évènement clavier :

```
ligne,colonne,état,code-touche,texte
Example :
1,1,N,0x41,texte
```

- Syntaxe définissant une touche exécutant une ou plusieurs commandes :

```
ligne,colonne,état,commande([paramètre]{,paramètre…}){;commande([paramètre]{,paramètre…})},texte
Examples :
1,1,N,SHOW_KEYBOARD(1,true),texte
1,1,N,OPEN('C:/Windows/');SHOW_KEYBOARD(0,false),texte
```

- Les commandes doivent être séparées par des « ; » ;
- Les paramètres des commandes doivent être séparés par des « , » ;
- Les paramètres de type chaîne de caractères doivent être en quotes ;
- Si l'on souhaite utiliser une quote dans une chaîne de caractères, il faut l'échapper avec un anti-slash « \ », exemple : « `'voici une quote \' échappée'` ».

Voir le paragraphe sur les commandes pour plus de détails.

- Syntaxe définissant une touche exécutant une ou plusieurs commandes et un évènement clavier :

```
ligne,colonne,état,code-touche{;commande([paramètre]{,paramètre…})},texte
Example :
1,1,N,0x41;SHOW_KEYBOARD(1,true),texte
1,1,N,0x41;OPEN('C:/Windows/');SHOW_KEYBOARD(0,false),texte
```

« ligne » et « colonne » indiquent la position de la touche sur la « grille » du clavier. « état » indique l'étaut du clavier au moment de l'émission de l'évènement clavier (« S » pour shift, « N » pour normal). La valeur de « code-touche » est facultative et indique le code de la touche dont l'appui doit être simulé, « texte » est le texte à afficher sur la touche, et enfin « lignes » et « colonnes », facultatifs, définissent la taille du bouton dont les dimensions sont exprimées en nombre de cases sur la grille du clavier virtuel.

2.2.2 Commandes.

Avec TouchKbd 3.1 il est possible d'effectuer des actions lors de l'appui d'une touche, en plus de l'émission d'un évènement clavier. Ces actions (ou commandes) sont effectuées par des fonctions prenant un certain nombre de données d'entrée qui permettent d'indiquer ce qu'elle doivent faire.

Il y a trois types de données d'entrée :

- « int » un nombre entier compris dans l'intervale [- 2,147,483,648 .. + 2,147,483,647] ;
- « bool » booléan ayant deux valeurs possibles : « true » ou « false », en minuscules ;
- « string » un chaîne de caractères encadrée par des quotes « ' » comme par exemple : « `'une chaîne de caractère d\'exemple contenant une quote'` ». Si la chaîne de caractère doit contenir une quote, celle-ci doit être échapée (précédée) par un anti-slash « \ ».

Les commandes disponibles sont les suivantes :

`SHOW_KEYBOARD``(int,bool)`

Permet d'afficher ou de masquer un clavier. Paramètres :
- Numéro du clavier (en partant de 0) ;
- Si il faut l'afficher (true), ou le masquer (false).

`SHOW_LAYOUT``(int,int,bool)`

Permet d'afficher ou de masquer toutes les touches d'une layout. Paramètres :
- Numéro du clavier (en partant de 0) ;
- Numéro de la layout (en partant de 0) ;
- Si il faut les afficher (true), ou les masquer (false).

`SHOW_KEY``(int,int,int,int,bool)`

Permet d'afficher ou de masquer une touche. Paramètres :
- Numéro du clavier (en partant de 0) ;
- Numéro de la layout (en partant de 0) ;
- Coordonées Y (ligne) de la touche (en partant de 0) ;
- Coordonées X (colonne) de la touche (en partant de 0) ;
- Si il faut l'afficher (true), ou la masquer (false).

ENABLE_KEYBOARD(int,bool)

Permet d'activer ou de désactiver toutes les touches d'un clavier. Paramètres :

- Numéro du clavier (en partant de 0) ;
- Si il faut les activer (true), ou les désactiver (false).

ENABLE_LAYOUT(int,int,bool)

Permet d'activer ou de désactiver toutes les touches d'une layout. Paramètres :

- Numéro du clavier (en partant de 0) ;
- Numéro de la layout (en partant de 0) ;
- Si il faut les activer (true), ou les désactiver (false).

ENABLE_KEY(int,int,int,int,bool)

Permet d'activer ou de désactiver une touche. Paramètres :

- Numéro du clavier (en partant de 0) ;
- Numéro de la layout (en partant de 0) ;
- Coordonées Y (ligne) de la touche (en partant de 0) ;
- Coordonées X (colonne) de la touche (en partant de 0) ;
- Si il faut l'activer cacher (true), ou la désactiver (false).

Les touches qui sont désactivées sont dessinées sans leur contour 3D et n'émètent aucun évènement clavier ni exécutent aucune action lors qu'elles sont appuyées.

SET_LAYOUT(int,int)

Permet de définir la layout active sur un clavier donné.

- Numéro du clavier (en partant de 0) ;
- Numéro de la layout (en partant de 0) .

EXIT()

Permet de fermer TouchKbd, ne prend aucun paramètre.

OPEN(string,string)

Permet d'ouvrir n'importe quel fichier (ou exécutable) ou dossier comme si l'on cliquait dessus dans l'explorateur de fichier, en ayant en plus la possibilité de passer des paramètres si le fichier est un exécutable. Si il n'y a aucun paramètre à passer, le deuxième paramètre doit être une chaine de caractères vide : « '' ».

- Chemin **absolu** vers le fichier ;
- Paramètres (si la cible est une exécutable) ou chaîne de caractères vide.

SET_KEY_BKG_COLOR(int,int,int,int,int,int,int)

Permet d'activer ou de désactiver une touche. Paramètres :

- Numéro du clavier (en partant de 0) ;
- Numéro de la layout (en partant de 0) ;
- Coordonnées Y (ligne) de la touche (en partant de 0) ;
- Coordonnées X (colonne) de la touche (en partant de 0) ;
- Quantité de rouge, entre 0 et 255 (format RGB) ;
- Quantité de vert, entre 0 et 255 (format RGB) ;
- Quantité de blue, entre 0 et 255 (format RGB).

Exemples :

```
1,1,S,0x41;OPEN('C:/Program Files/Mozilla Firefox/firefox.exe','http://www.google.com/'),A
```

Définit une touché en position (1;1) qui émet un évènement clavier « shift+a » (donc A majuscule) et qui lance le navigateur Firefox avec en paramètre l'adresse de la page à ouvrir.

```
1,1,N,0x42,b,2,2
```

Définit une touche en position (1;1) qui émet un évènement clavier « b » et qui a une taille de 2x2.

```
1,1,N,,vide¤1
```

Touche qui n'émet aucun évènement ni exécute aucune action, qui affiche le texte « vide » et qui a pour style le style numéro 1.

```
1,1,N,EXIT(),Sortir
```

Touche qui ferme TouchKbd sans émettre aucun évènement clavier.

3 TouchKbd 3.0

3.1 Gestion du multi-clavier

Cette fonctionnalité doit permettre d'afficher plusieurs claviers, chacun ayant son propre ensemble de layouts. La limite du nombre de claviers est pour l'instant fixée à 5 mais peut être modifiée à la compilation du programme.

Pour le premier clavier, la syntaxe du fichier de configuration reste la même. Un bloc « [TOUCH_KBD] » pour paramétrer le premier clavier et sa première layout, puis un bloc « [TOUCH_KBD_L] » par layout supplémentaire où L est le numéro de la layout en partant de 0.

```
[TOUCH_KBD]
Paramétrage du clavier
Définition de la première layout

[TOUCH_KBD_1]
Définition de la deuxième layout

[TOUCH_KBD_2]
Définition de la troisème layout
```

Si l'on souhaite ajouter un clavier, il faudra :

- Ajouter dans le bloc « [TOUCH_KBD] » un champ « Keyboards » ayant pour valeur le nombre de claviers. Exemple : « Keyboards=3 » . Si ce champ n'est pas spécifié, les claviers supplémentaires seront ignorés. Si sa valeur est incohérente, le programme s'arrêtera.
- Ajouter un bloc « TOUCH_KBD_EXT_C] » où C est le numéro du clavier en partant de zéro. Tout comme le bloc « [TOUCH_KBD] » ce bloc définira les paramètres du clavier ainsi qu'une première layout. Attention : ne pas oublier le champ « Layouts » qui définit le nombre de layout d'un clavier lors de l'ajout de claviers supplémentaires.

Si l'on souhaite ajouter des layouts supplémentaires à un clavier supplémentaire, il suffit d'ajouter un bloc « [TOUCH_KBD_EXT_C_L] » où C est le numéro du clavier, et L la numéro de la layout, toujours en partant de zéro. Exemple :

```
[TOUCH_KBD]
Paramétrage du clavier (numéro 0)
Définition de la première layout

[TOUCH_KBD_1]
Définition de la deuxième layout

{TOUCH_KBD_EXT_1]
Paramétrage du second clavier (numéro 1)
Définition de la première layout

[TOUCH_KBD_EXT_1_1]
Définition de la deuxième layout
```

3.2 Ajout de styles

Afin de pouvoir personnaliser l'apparence du clavier Il doit être possible d'ajouter à style à chaque clavier, layout, et bouton.

Pour le clavier, il faut ajouter un champ nommé « Style » ayant pour valeur le numéro du style à appliquer. Par exemple :

```
[TOUCH_KBD]                                          Style 0
Paramétrage du clavier (numéro 0)
Style=0
Définition de la première layout

[TOUCH_KBD_1]                                        Style 1, prévaut sur celui du clavier
Définition de la deuxième layout
Style=1

{TOUCH_KBD_EXT_1]                                    Style 3
Paramétrage du second clavier (numéro 1)
Style=3
Définition de la première layout

[TOUCH_KBD_EXT_1_1]                                  Style non défini, style 3 hérité du clavier
Définition de la deuxième layout
```

Le style d'un bouton prévaut sur celui de se layout, et celui d'une layout prévaut sur celui de son clavier. Par exemple, si un clavier à un style définissant la couleur de la police comme rouge, mais qu'une touche a un style la définissant comme bleue, alors la touche aura une police bleue.

Lorsqu'une propriété (couleur, taille, etc) n'est pas définie, par exemple si une touche a un style qui ne définit pas de couleur de police, alors celle définie par le style de la layout (si il y en a un) sera utilisée, ou à défaut celle du style du clavier (si il y en a un), ou à défaut une couleur par défaut.

Pour attribuer un style à une touche, il faut ajouter le numéro du style après le texte à afficher sur la touche en utilisant le caractère séparateur « ¤ ». Par exemple :

```
Key01=1,1,N,0x70,F1¤3
Key02=1,1,N,0x70,F1¤2,1,2
```

3.3 Définition d'un style

Les styles doivent être définis dans le bloc « [TOUCH_KBD_STYLES] ». Les propriétés que l'on ne souhaite pas définir doivent tout de même y figurer, mais avec une chaîne de caractères vide.

Chaque style peut définir un ensemble de propriétés. Les propriétés pouvant être définies sont les suivantes :
- Le nom de la police (FontName) : peut contenir des espaces ;
- La taille de la police (FontSize) : en nombre de pixels ;
- La couleur de la police (FontColorRGB) : au format RGB, avec des valeurs séparées par des « , » ;
- La couleur de fond (BkgColorRGB) : au même format que la couleur de la police ;
- Un dégradé de fond (BkgGradient) : syntaxe décrite plus loin ;
- L'image de fond (BkgImageSrc) : chemin absolu vers le fichier. Images bitmap « .bmp » uniquement.
- Si le text est en gras (FontBold) : « Yes » ou « No » ;
- Si le text est en italique (FontItalic) : « Yes » ou « No » ;
- Si le text est souligné (FontUnderlined) : « Yes » ou « No » ;

Tous les styles étant définis dans la même section, leurs propriétés doivent être suivies du numéro de leur style afin de pouvoir relier les y relier. Par exemple :

```
[TOUCH_KBD_STYLES]
// Style principal
BkgImageSrc0=
BkgColorRGB0=150,150,255
BkgGradient0=255,0,0,100%/0,255,0,100%[50%]/0,0,255,100%
FontName0=Times new roman
FontSize0=
FontColorRGB0=0,0,50
FontBold0=Yes
FontItalic0=No
FontUnderlined0=No
// Style flèches
BkgImageSrc2=
BkgColorRGB2=200,200,255
BkgGradient2=
FontName2=Wingdings
FontSize2=32
FontColorRGB2=0,0,50
FontBold2=Yes
FontItalic2=
FontUnderlined2=

[TOUCH_KBD]
Style=0
Layouts=1
Keyboards=1
// Etc...
```

Pour plus de lisibilité, il est possible de séparer les groupes de propriétés par des commentaires (ligne commençant par « // » et contenant n'importe quel texte) où l'on pourra écrire, par exemple, le nom d'un style.

3.3.1 Syntaxe des dégradés

Les dégradés sont définis par un ensemble « d'étapes ». Chaque étape définit la couleur que l'on doit avoir à une position donnée, et le dégradé se fait entre chacune de ces étapes. Le nombre d'étapes maximal pour un même dégradé est fixé à 10.

Par exemple, pour un dégradé blanc vers noir (au milieu) vers blanc, on aura trois étapes : blanc à la position 0%, noir à la position 50%, et à nouveau blanc à la position 100%.

Le sens du dégradé est indiqué par le premier caractère : H pour horizontal, et V pour vertical.

Quelques règles :

- Couleurs : exprimées au format RGB, de 0 à 255 par canal ;
- Opacité (alpha) : en pourcentage, de 0% à 100% ;
- Position : en pourcentage, de 0% à 100%.
- La position de la première étape et celle de la dernière sont toujours de 0% et 100% respectivement, il est donc inutile de les préciser.
- Les étapes doivent être séparées par des « / » ;
- Les canaux de couleurs doivent être séparés par des « ; » ;
- Les positions sont écrites entre crochets ;
- La définition ne doit pas contenir de caractères d'espaces.

La syntaxe est la suivante :

```
H/rouge,vert,bleu,alpha%/{rouge,vert,bleu,alpha%[position%]/}rouge,vert,bleu,alpha%
```

Où {contenu} indique entre 0 et N fois le contenu.

Pour l'exemple décrit précédemment on aura donc :

```
H/255;255;255;100%/0;0;0;100%[50%]/255;255;255;100%
```

3.4 Gestion de la variable « KeyCount »

Dans chaque bloc définissant une layout il doit y avoir un champ « KeyCount » indiquant le nombre de touches définies pour cette « layout ». Cette variable doit être prise en compte lors de la lecture du fichier de configuration.

3.5 Messages externes

Il est désormais possible pour un programme externe (comme le Telnet par exemple) d'envoyer des messages à TouchKbd pour lui commander de faire certaines actions. Les messages actuellement gérés sont les suivants :

- **WM_TKBD_ACTIVATE** :
  - Valeur : WM_USER + 5713
  - Valeurs pour WPARAM :
    - **WP_TKBD_SHOW** :
      - Valeur : 1
      - Description : Affiche TouchKbd s'il est actuellement masqué. Si TouchKbd a été masqué par un message WM_TKBD_ACTIVATE, alors seuls les claviers précédemment affichés sont réaffichés. Si tous les claviers ont étés masqués par TouchKbd, alors la visibilité initiale de chaque clavier est restaurée.
    - **WP_TKBD_HIDE** :
      - Valeur : 2
      - Description : Masque TouchKbd s'il est actuellement visible. La visibilité de chaque de chaque clavier au moment du masquage est sauvegardée pour être restaurée lors du réaffichage.
    - **WP_TKBD_TOGGLE** :
      - Valeur : 3
      - Description : Même comportement que WP_TKBD_SHOW ou WP_TKBD_HIDE selon que TouchKbd est masqué ou non.
    - **WP_TKBD_SET_LAYOUT** :
      - Valeur : 4
      - Description : Définit la layout courrante d'un clavier donné. Le LPARAM doit contenir les deux index : le « high word » contient l'index du du clavier et le « low word » celui de la layout. Les index commencent à 0.
    - **WP_TKBD_PREV_LAYOUT** :
      - Valeur : 5
      - Description : Passe à la layout précédente (selon l'ordre de définition dans la config) dans le clavier donné. Le « high word » du LPARAM doit contenir l'index du clavier. Les index commencent à 0.
    - **WP_TKBD_NEXT_LAYOUT** :
      - Valeur : 6
      - Description : Passe à la layout suivante (selon l'ordre de définition dans la config) dans le clavier donné. Le « high word » du LPARAM doit contenir l'index du clavier. Les index commencent à 0.
    - **WP_TKBD_SHOW_KEYBOARD** :
      - Valeur : 7
      - Description : Affiche le clavier donné s'il ne l'est pas. Le « high word » du LPARAM doit contenir l'index du clavier. Les index commencent à 0.

- **WP_TKBD_HIDE_KEYBOARD** :
  - Valeur : 8
  - Description : Masque le clavier donné s'il est affiché. Le « high word » du LPARAM doit contenir l'index du clavier. Les index commencent à 0.
  -

Cette fonctionnalité nécessite la présence, dans la section « TOUCH_KBD » du registre, d'une valeur nommée « HWND » de type DWORD et ayant pour valeur le « handle » de la fenêtre principale de TouchKbd. Cette valeur est initialisée et supprimée à l'ouverture et à la fermeture de TouchKbd (respectivement). Cette valeur est nécessaire aux programmes externes souhaitant intéragir avec TouchKbd pour savoir à quelle fenêtre envoyer ces messages.