Qu'est-ce que le Terminal Windows ?

Article • 24/05/2023

Terminal Windows est une application hôte moderne pour les interpréteurs en ligne de commande que vous appréciez déjà, tels que l'invite de commandes, PowerShell et bash (via Sous-système Windows pour Linux (WSL)). Ses principales fonctionnalités comprennent un affichage multi-onglet, des volets, une prise en charge des caractères Unicode et UTF-8, un moteur de rendu de texte accéléré par GPU, ainsi que la possibilité de créer vos propres thèmes et de personnaliser le texte, les couleurs, les arrière-plans et les touches de raccourci.

Installer le Terminal Windows

https://www.microsoft.com/fr-fr/videoplayer/embed/RWHAdS?postJsllMsg=true&autoCaptions=fr-fr

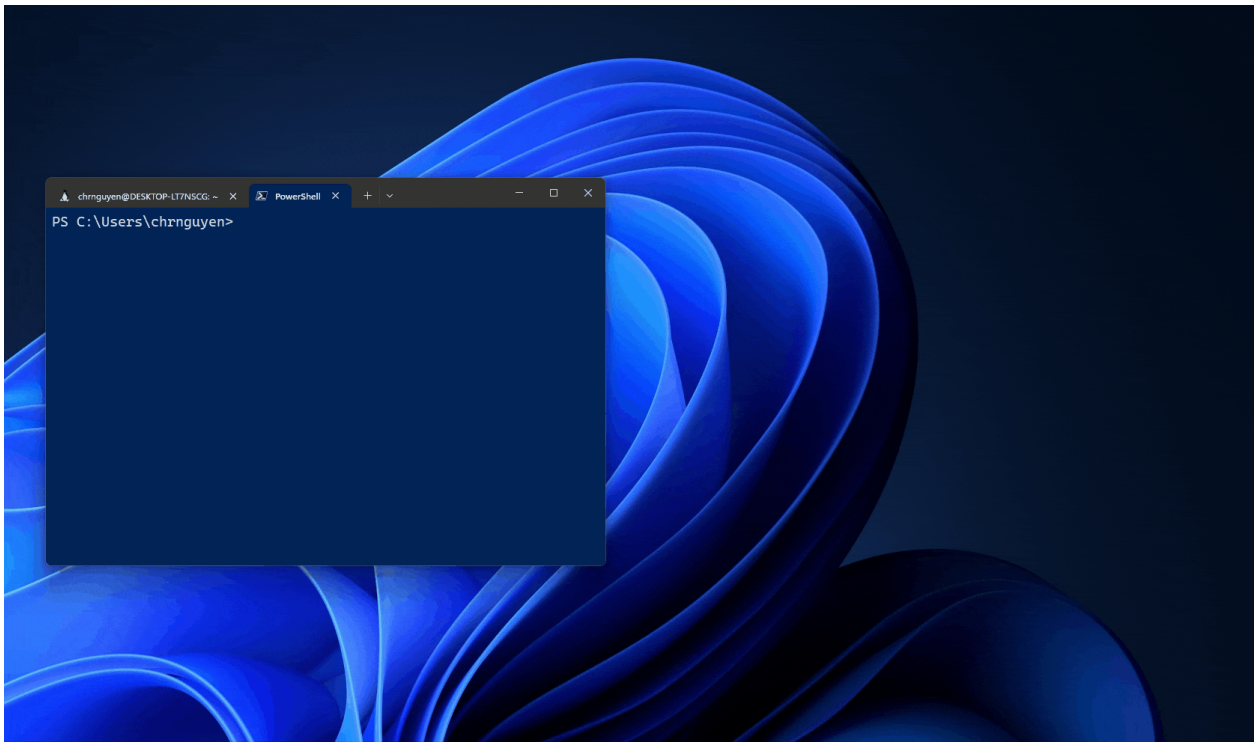
> **Notes**
>
> Pour plus d'informations générales, consultez l'article de Scott Hanselman, **What's the difference between a console, a terminal, and a shell?**, ou la vidéo de Rich Turner, **What is a command-line shell?**.

Plusieurs profils prenant en charge une grande variété d'applications de ligne de commande

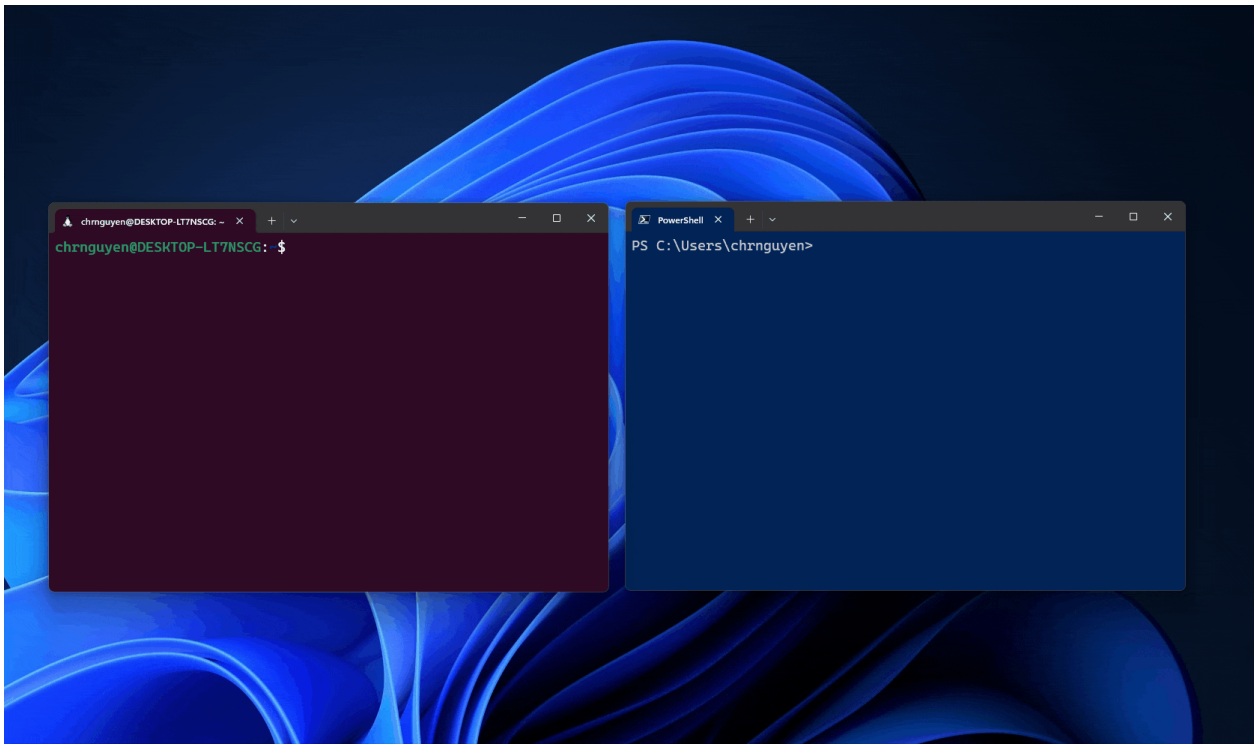
Toute application dotée d'une interface de ligne de commande peut être exécutée à l'intérieur de Terminal Windows. Cela inclut tous les éléments de PowerShell et de l'invite de commandes pour Azure Cloud Shell et toute distribution WSL telle qu'Ubuntu ou Oh-My-Zsh.

Détachement d'onglets (Préversion)

Dans Terminal Windows, vous pouvez détacher des onglets d'une fenêtre pour les mettre dans de nouvelles fenêtres.



Vous pouvez également glisser-déposer des onglets dans des fenêtres existantes.



> **ⓘ Important**
>
> Cette fonctionnalité n'est disponible que dans la préversion de Terminal Windows.

Schémas et configurations personnalisés

Vous pouvez configurer votre Terminal Windows pour disposer d'un large éventail de modèles de couleurs et de paramètres. Pour savoir comment personnaliser votre invite avec des thèmes sympa, consultez Tutoriel : Configurer une invite personnalisée pour PowerShell ou WSL avec Oh My Posh Pour savoir comment créer votre propre jeu de couleurs, visitez la page Jeux de couleurs.

Actions personnalisées

Il existe une variété de commandes personnalisées que vous pouvez utiliser dans Terminal Windows pour qu'il vous semble plus naturel. Si vous n'aimez pas un raccourci clavier particulier, vous pouvez le remplacer par ce que vous préférez.

Par exemple, le raccourci par défaut pour copier le texte à partir de la ligne de commande est `Ctrl + Maj + c`. Vous pouvez le remplacer par `Ctrl + 1` ou par toute autre combinaison que vous préférez. Pour ouvrir un nouvel onglet, le raccourci par défaut est `Ctrl + Maj + t`, mais peut-être souhaitez-vous le remplacer par `Ctrl + 2`. Le raccourci par défaut pour basculer entre les onglets ouverts est `Ctrl + Tab`, mais il peut être remplacé par `Ctrl + -` et utilisé pour créer un nouvel onglet à la place.

Vous pouvez découvrir la personnalisation des raccourcis dans la page Actions.

Prise en charge des caractères Unicode et UTF-8

Terminal Windows peut afficher des caractères Unicode et UTF-8, tels que des Emoji et des caractères issus de différentes langues.

Rendu du texte accéléré par GPU

Terminal Windows utilise le GPU pour restituer du texte, ce qui améliore les performances par rapport à l'expérience de ligne de commande Windows par défaut.

Prise en charge des images d'arrière-plan

Vous pouvez avoir des images d'arrière-plan et des gif dans votre fenêtre Terminal Windows. Vous trouverez des informations sur l'ajout d'images d'arrière-plan à votre profil sur la page Profil - Apparence.

Arguments de ligne de commande

Vous pouvez définir Terminal Windows pour le lancer dans une configuration spécifique à l'aide d'arguments de ligne de commande. Vous pouvez spécifier le profil à ouvrir dans un nouvel onglet, le répertoire de dossiers à sélectionner, ouvrir le terminal à l'aide de volets de fenêtre fractionnée, puis choisir l'onglet qui doit être actif.

Par exemple, pour ouvrir Terminal Windows à partir de PowerShell avec trois volets, le volet gauche exécutant un profil d'invite de commandes et le volet droit étant réparti entre votre PowerShell et votre profil par défaut exécutant WSL, entrez :

PowerShell

```
wt -p "Command Prompt" `; split-pane -p "Windows PowerShell" `; split-pane -
H wsl.exe
```

Découvrez comment configurer les arguments de ligne de commande à la page Arguments de ligne de commande.

Installer le Terminal Windows et commencer à le configurer

Article • 30/05/2023

Installer

Installer le Terminal Windows

Pour tester les dernières fonctionnalités d'évaluation, vous pouvez également installer le Terminal Windows (préversion).

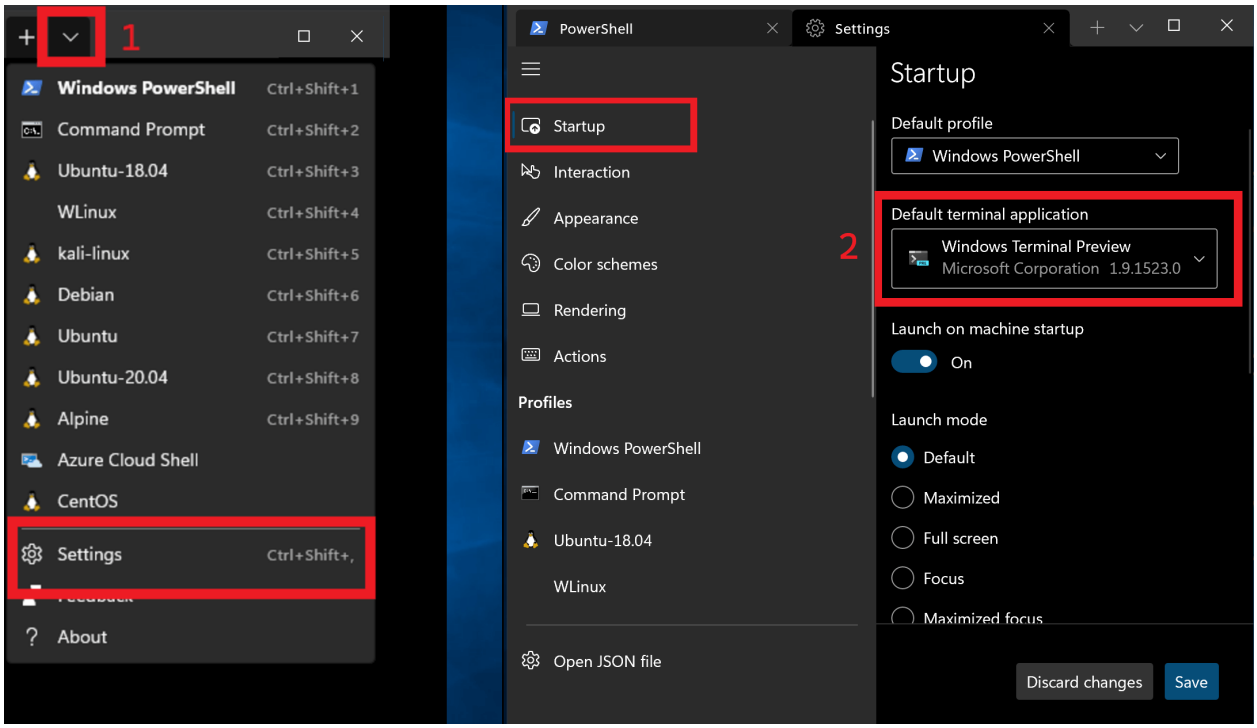
> **Notes**
>
> Si vous n'avez pas accès au Microsoft Store, les builds sont publiées sur la **page des versions dans GitHub**. Si vous effectuez l'installation à partir de GitHub, le Terminal Windows n'est pas automatiquement mis à jour avec les nouvelles versions. Pour utiliser un gestionnaire de package (WinGet, Chocolatey, Scoop) et bénéficier d'options d'installation supplémentaires, consultez le **dépôt du produit Terminal Windows**.

Définir votre application de terminal par défaut

> **Notes**
>
> Cette fonctionnalité est disponible dans toutes les versions de Windows 11 et de Windows 10 22H2 après l'installation de la mise à jour du 23 mai 2023, **KB5026435**.

Pour ouvrir une application de ligne de commande avec le Terminal Windows, définissez-la comme application de terminal par défaut.

1. Ouvrez le Terminal Windows et accédez à la fenêtre **Paramètres** dans l'interface utilisateur.
2. Sélectionnez **Démarrage**, puis choisissez « Terminal Windows » comme **Application de terminal par défaut**.



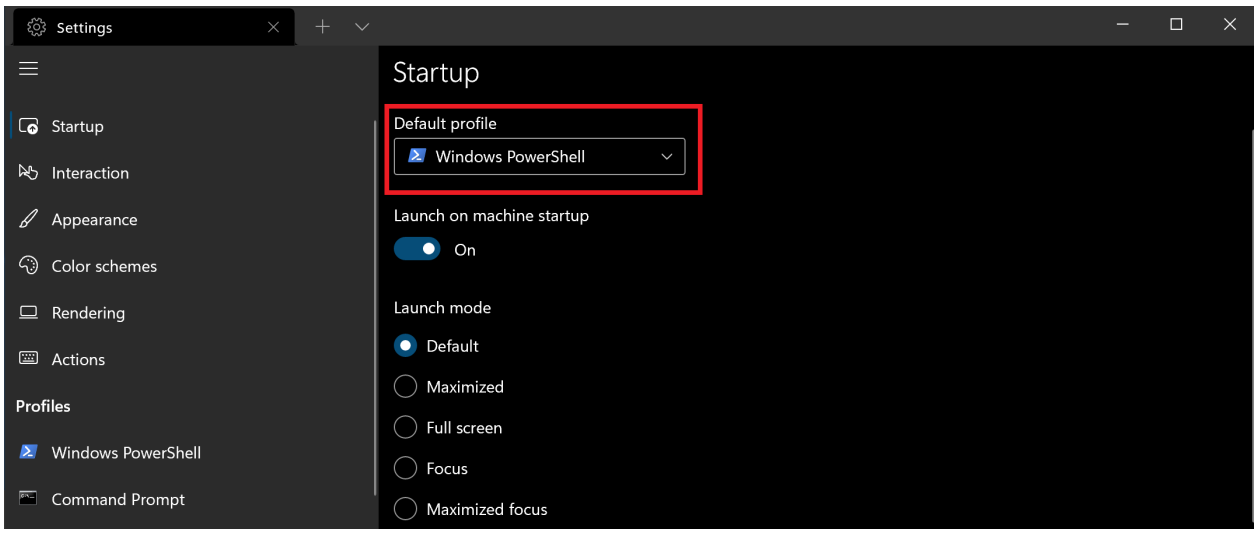
Définir votre profil de terminal par défaut

Une fois l'installation terminée, quand vous ouvrez le Terminal Windows, il démarre avec la ligne de commande PowerShell comme profil par défaut sous l'onglet ouvert.

Pour modifier le profil par défaut :

1. Ouvrez le Terminal Windows et accédez à la fenêtre **Paramètres** dans l'interface utilisateur.
2. Sélectionnez **Démarrage** et choisissez le **Profil par défaut** désiré.

Vous pouvez également définir votre profil par défaut dans le fichier Settings.json associé au Terminal Windows.

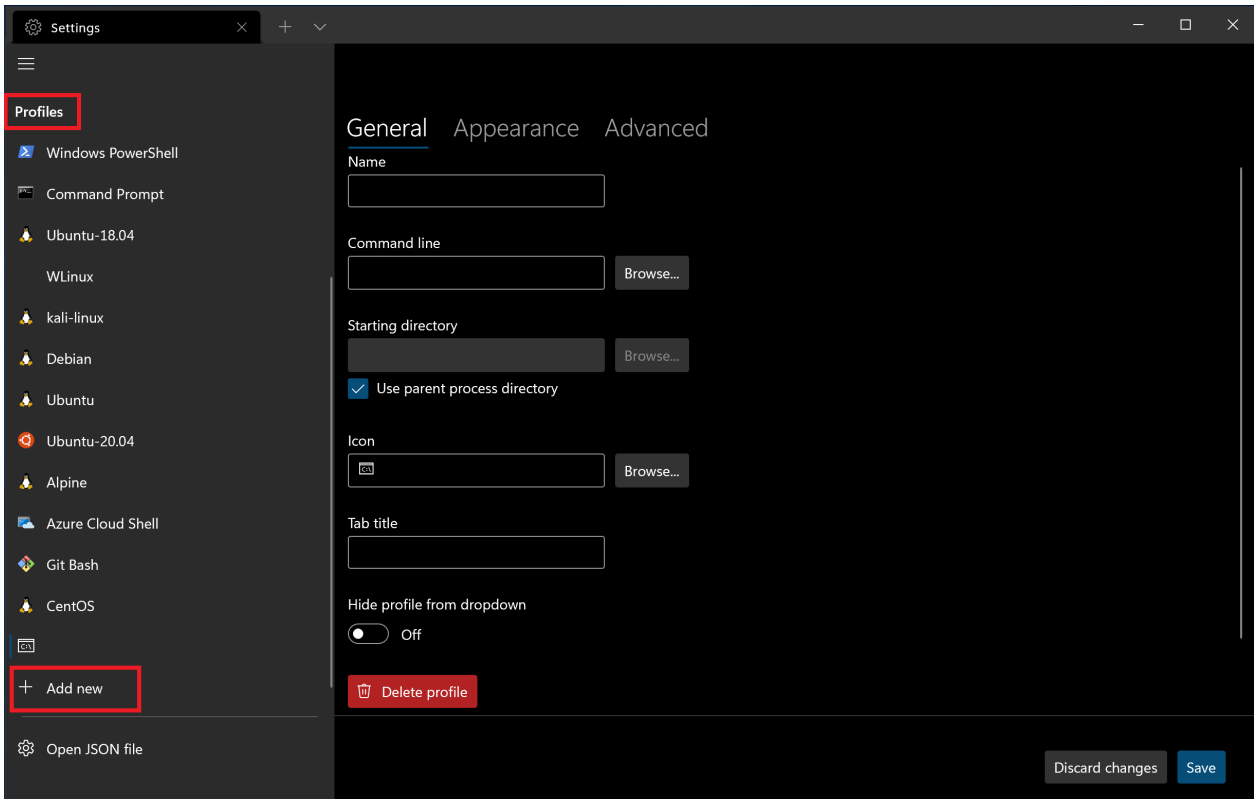


Ajouter de nouveaux profils

Le Terminal Windows crée automatiquement des profils pour vous si des distributions WSL ou plusieurs versions de PowerShell sont installées.

Vos profils de ligne de commande sont listés dans la fenêtre Paramètres de l'interface utilisateur, qui contient également une option + **Ajouter nouveau** pour ajouter de nouveaux profils.

En savoir plus sur les profils dynamiques à la page Profils dynamiques.

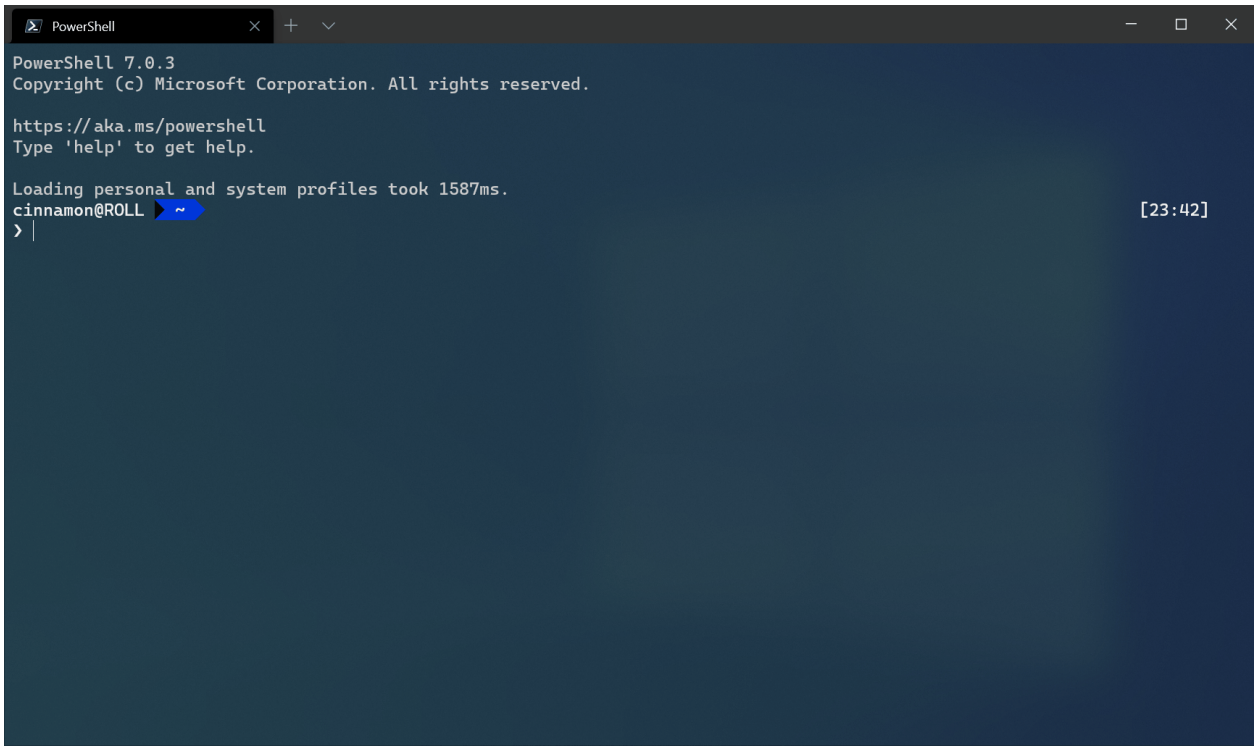


Ouvrir un nouvel onglet

Vous pouvez ouvrir un nouvel onglet du profil par défaut en appuyant sur `Ctrl`+`Maj`+`T` ou en sélectionnant le bouton + (plus). Pour ouvrir un autre profil, sélectionnez la touche ∨ (flèche) en regard du bouton + pour ouvrir le menu déroulant. De là, vous pouvez sélectionner le profil à ouvrir.

Appeler la palette de commandes

Vous pouvez appeler la plupart des fonctionnalités du Terminal Windows par le biais de la palette de commandes. La combinaison de touches par défaut à appeler est `Ctrl`+`Maj`+`P`. Vous pouvez également l'ouvrir en sélectionnant le bouton **Palette de commandes** situé dans le menu déroulant du Terminal Windows (préversion).



Ouvrir un nouveau volet

Vous pouvez exécuter plusieurs interpréteurs de commandes côte à côte à l'aide des volets. Pour ouvrir un volet, vous pouvez utiliser `Alt`+`Maj`+`+` pour un volet vertical ou `Alt`+`Maj`+`-` pour un volet horizontal. Vous pouvez aussi utiliser `Alt`+`Maj`+`D` pour ouvrir un volet dupliqué de votre profil qui a le focus. En savoir plus sur les volets à la page Volets.

Configuration

Pour personnaliser les paramètres de votre Terminal Windows, sélectionnez **Paramètres** dans le menu déroulant. L'interface utilisateur des paramètres s'ouvre pour vous permettre de configurer vos paramètres. Vous pouvez apprendre à ouvrir l'interface utilisateur des paramètres en utilisant les raccourcis clavier dans la page Actions.

Fichier de paramètres JSON

Si vous préférez configurer les paramètres du Terminal Windows avec du code plutôt que d'utiliser l'interface utilisateur graphique, vous pouvez modifier le fichier settings.json.

Sélectionnez **Paramètres** dans le menu déroulant du Terminal Windows tout en maintenant la touche `Maj` enfoncée pour ouvrir le fichier `settings.json` dans votre

éditeur de texte par défaut. (L'éditeur de texte par défaut est défini dans vos paramètres Windows.)

Le fichier settings.json du Terminal Windows peut se trouver dans l'un des répertoires suivants :

- Terminal (version stable/générale) : `%LOCALAPPDATA%\Packages\Microsoft.WindowsTerminal_8wekyb3d8bbwe\LocalState\settings.json`
- Terminal (préversion) : `%LOCALAPPDATA%\Packages\Microsoft.WindowsTerminalPreview_8wekyb3d8bbwe\LocalState\settings.json`
- Terminal (non packagé : Scoop, Chocolatey, etc.) : `%LOCALAPPDATA%\Microsoft\Windows Terminal\settings.json`

> **Conseil**
>
> 1. Pour accéder aux paramètres par défaut du Terminal Windows, sélectionnez **Paramètres** dans le menu déroulant tout en maintenant la touche `Alt` enfoncée. Le fichier `defaults.json` s'ouvre dans votre éditeur de texte par défaut. Ce fichier est généré automatiquement et toute modification est ignorée.
> 2. Il est possible de créer une **extension de fragment JSON** pour stocker les données et les modèles de couleurs du profil dans un fichier distinct, ce qui peut être utile pour éviter des fichiers de configuration de trop grande taille.

Arguments de ligne de commande

Vous pouvez lancer le terminal dans une configuration spécifique à l'aide d'arguments de ligne de commande. Ces arguments vous permettent d'ouvrir le terminal avec des onglets et des volets spécifiques ainsi que des paramètres de profil personnalisés. En savoir plus sur les arguments de ligne de commande à la page Arguments de ligne de commande.

Résolution des problèmes

Si vous rencontrez des difficultés à utiliser le terminal, reportez-vous à la page Résolution des problèmes. Si vous trouvez des bogues ou que vous avez une demande concernant la fonctionnalité, vous pouvez sélectionner le lien de commentaires dans le

menu **À propos de** du terminal pour accéder à la page GitHub, où vous pouvez signaler un nouveau problème.

Types de distribution Terminal Windows

Article • 24/05/2023

Terminal Windows est distribué via des versions GitHub dans différents formats :

- Empaqueté ou sous « bundle MSIX »
  - Il s'agit de la distribution de Terminal Windows la plus ancienne et la mieux prise en charge.
  - La distribution empaquetée peut être installée via le fichier `.msixbundle` fourni dans la page des versions GitHub ou via le Microsoft Store (Stable, Préversion).
  - L'installation via un bundle MSIX peut nécessiter une connectivité réseau pour télécharger les packages de dépendances à partir du Store.
  - Lorsqu'il est installé via le bundle MSIX, Terminal reçoit des mises à jour automatiques via le Store.
- Kit de préinstallation
  - Un kit de préinstallation est disponible pour les intégrateurs système et les OEM qui souhaitent préinstaller Terminal Windows sur une image Windows.
  - Vous trouverez plus d'informations dans la documentation DISM sur la préinstallation. Les utilisateurs qui n'ont pas l'intention de préinstaller Terminal Windows doivent continuer à utiliser la distribution empaquetée.
  - Lorsqu'il est installé via le kit de préinstallation, Terminal reçoit des mises à jour automatiques via le Store.
- Non empaqueté ou « ZIP » (nouveau dans la version 1.17 stable)
  - Cette méthode de distribution n'a pas été officiellement prise en charge jusqu'à la version de canal 1.17 stable.
  - La distribution non empaquetée ne reçoit pas de mises à jour automatiques, ce qui vous permet de contrôler exactement quand les nouvelles versions sont installées.
- Portable
  - Variante de la distribution non empaquetée, où Terminal stocke ses paramètres dans un répertoire à proximité.
  - En savoir plus sur la configuration du mode portable.

Comparaison des fonctionnalités de distribution

	Empaqueté	Kit de préinstallation	Non empaqueté	Portable

	Empaqueté	Kit de préinstallation	Non empaqueté	Portable
Mises à jour automatiques	✅	✅	❌	❌
Sélection automatique de l'architecture	✅	✅	❌	❌
Peut être défini comme Terminal par défaut	✅	✅	❌	❌
Menu contextuel « Ouvrir dans Terminal »	✅	✅	❌	❌
Option de démarrage automatique à la connexion	✅	✅	*manual*	*manual*
Installation en un double-clic	✅	❌	❌	❌
Installation sur des ordinateurs qui ne sont pas en réseau	❌	✅	✅	✅
Préinstallation dans une image Windows	❌	✅	*en tant que fichiers bruts*	*en tant que fichiers bruts*
Chemin d'installation contrôlé par l'utilisateur	❌	❌	✅	✅
Double-clic activable	❌	❌	✅	✅
Emplacement de stockage des paramètres	Dossier utilisateur, par paquet	(identique à empaqueté)	`%LOCALAPPDATA%`	À côté de `WindowsTerminal.exe`

Terminal Windows en mode portable

À compter de la version de canal 1.17 stable, Terminal Windows peut être déployé en « mode portable ». Le mode portable garantit que toutes les données créées et

gérées par Terminal Windows sont enregistrées à côté de l'application afin qu'elles puissent être déplacées plus facilement entre différents environnements.

Le mode portable est pris en charge par la distribution « ZIP » non empaquetée.

Il s'agit d'un mode d'exécution officiellement pris en charge où Terminal Windows stocke ses paramètres dans un dossier `settings` à côté de `WindowsTerminal.exe`.

Le mode portable n'est pas pris en charge dans les distributions empaquetées ou de kit de préinstallation de Terminal Windows.

Pourquoi utiliser le mode portable ?

Les distributions non empaquetées et en mode portable de Terminal Windows vous permettent d'utiliser Terminal sans l'installer globalement, c'est-à-dire sur des systèmes où vous n'avez peut-être pas l'autorisation d'installer des packages MSIX ou de télécharger des logiciels à partir du Microsoft Store.

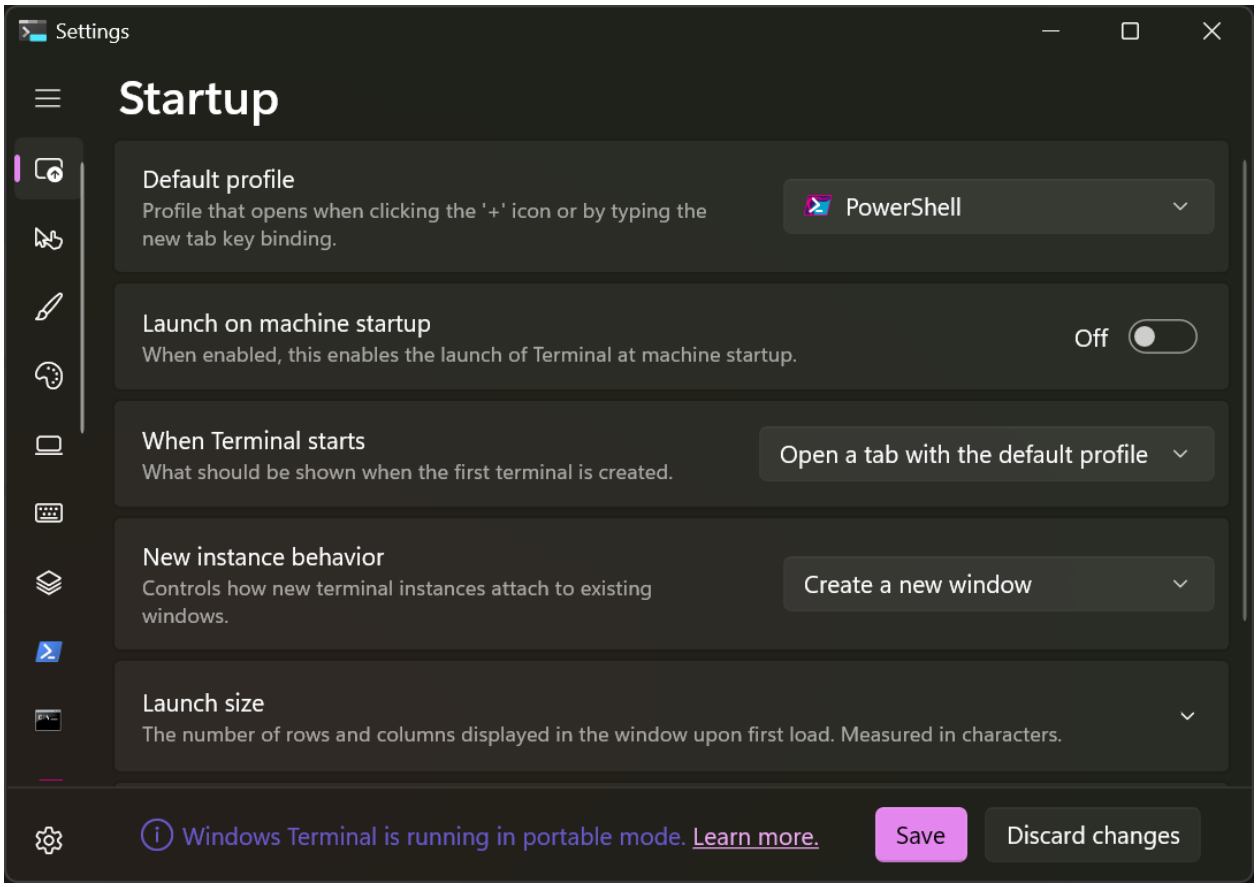
Le mode portable vous permet de transporter ou d'archiver une installation préconfigurée de Terminal Windows et de l'exécuter à partir d'un partage réseau, d'un lecteur cloud ou d'une clé USB. Ce type d'installation est autonome et n'interfère pas avec d'autres distributions installées de Terminal Windows.

Activation du mode portable

Le mode portable doit être activé manuellement. Après avoir décompressé le téléchargement de Terminal Windows, créez un fichier nommé `.portable` à côté de `WindowsTerminal.exe`.

> ⓘ **Notes**
>
> Terminal Windows ne recharge pas automatiquement ses paramètres lorsque vous créez le fichier marqueur du mode portable. Cette modification s'applique uniquement après avoir relancé Terminal.

Terminal Windows crée automatiquement un répertoire nommé `settings` dans lequel il stocke à la fois les paramètres et l'état d'exécution comme les dispositions de fenêtre.



Désactivation du mode portable

Vous pouvez restaurer l'installation non empaquetée du mode portable à sa configuration d'origine, où les paramètres sont stockés dans `%LOCALAPPDATA%\Microsoft\Windows Terminal` en supprimant le fichier marqueur `.portable` du répertoire contenant `WindowsTerminal.exe`.

Si vous souhaitez réactiver le mode portable, vous pouvez créer un fichier marqueur `.portable` à côté de `WindowsTerminal.exe`.

Mise à niveau d'une installation en mode portable

Vous pouvez mettre à niveau une installation en mode portable de Terminal Windows en déplaçant le fichier marqueur `.portable` et le répertoire `settings` vers une version non empaquetée de Terminal Windows qui vient d'être extraite.

Paramètres de démarrage dans le Terminal Windows

Article • 21/03/2023

Les propriétés listées ci-dessous affectent tout le Terminal Windows, quels que soient les paramètres du profil. Elles doivent être placées à la racine de votre fichier settings.json.

Profil par défaut

Définissez le profil par défaut qui s'ouvre en tapant `Ctrl + Maj + t`, en tapant la combinaison de touches affectée à `newTab`, en exécutant `wt new-tab` sans spécifier de profil, ou en cliquant sur l'icône « + ».

Nom de la propriété : `defaultProfile`

Nécessité : Obligatoire

Accepte : GUID ou nom de profil sous forme de chaîne

Valeur par défaut : GUID de PowerShell

Application de terminal par défaut

Définissez l'émulateur de terminal par défaut dans Windows où toutes les applications de ligne de commande sont exécutées.

Nom de la propriété : Modifie un paramètre du système d'exploitation (sans nom de propriété dans le fichier settings.json).

Nécessité : Obligatoire

Accepte : Tout émulateur de terminal qui apparaît dans la liste déroulante

valeur par défaut : Hôte de console Windows

> ⓘ **Important**
>
> Cette fonctionnalité est disponible uniquement si vous exécutez le programme Insider de Windows 10 (Canal développeurs) ou Windows 11.

Lancer au démarrage de l'ordinateur

Quand la valeur est `true`, cela permet le lancement de Terminal Windows au démarrage. Si la valeur est `false`, l'entrée de la tâche de démarrage est désactivée.

Remarque : si l'entrée de la tâche de démarrage de Terminal Windows est désactivée par la stratégie d'organisation ou par l'action de l'utilisateur, ce paramètre n'a aucun effet.

Nom de la propriété : `startOnUserLogin`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Comportement lors du démarrage d'une nouvelle session de terminal

Si la valeur est `"defaultProfile"`, le Terminal Windows démarre une nouvelle session et ouvre un onglet avec votre profil par défaut.

Si la valeur est `"persistedWindowLayout"`, le Terminal Windows enregistre la disposition des fenêtres ouvertes lors de la fermeture et restaure toutes les fenêtres enregistrées lors du démarrage d'une nouvelle session. Le Terminal Windows enregistre automatiquement la disposition de toutes les fenêtres ouvertes pour faciliter la restauration après un plantage. Il enregistre également la mise en page si vous utilisez l'action `quit`. Par ailleurs, si vous fermez la dernière fenêtre ouverte en cliquant sur le bouton `X` ou en utilisant la commande `closeWindow`, il enregistre la disposition de cette dernière fenêtre.

Remarque : Le Terminal Windows enregistre actuellement les informations suivantes.

- Position, taille et nom de chaque fenêtre
- Disposition des onglets de chaque fenêtre, notamment la disposition et le profil de chaque volet, mais pas le contenu de ces volets

- Si votre shell est configuré pour signaler le répertoire de travail actif qui sera également enregistré

Nom de la propriété : `firstWindowPreference`

Nécessité : Facultatif

Accepte : `"defaultProfile"`, `"persistedWindowLayout"`

Valeur par défaut : `"defaultProfile"`

Mode de lancement

Le mode de lancement définit si le terminal est lancé en mode agrandi, en plein écran ou dans une fenêtre. L'affectation de la valeur `focus` équivaut à lancer le terminal en mode `default`, mais avec le mode Focus activé. De même, l'affectation de la valeur `maximizedFocus` entraîne le lancement du terminal dans une fenêtre agrandie avec le mode Focus activé.

Nom de la propriété : `launchMode`

Nécessité : Facultatif

Accepte : `"default"`, `"maximized"`, `"fullscreen"`, `"focus"`, `"maximizedFocus"`

Valeur par défaut : `"default"`

Comportement des nouvelles instances

Ce paramètre contrôle la manière dont les nouvelles instances de terminal s'attachent à des fenêtres existantes. Cette propriété est utilisée seulement si l'`--window,-w window`argument de ligne de commande n'est pas fourni. Ce paramètre accepte les valeurs possibles suivantes :

- `useNew` : crée toujours une fenêtre. Il s'agit du comportement standard du terminal avant la version 1.7.
- `useExisting` : crée des onglets dans la dernière fenêtre utilisée sur ce bureau. Si ce bureau virtuel ne compte aucune fenêtre existante, une fenêtre de terminal est créée.

- `useAnyExisting` : crée des onglets dans la dernière fenêtre utilisée, quel que soit le bureau virtuel où se trouve la fenêtre.

Nom de la propriété : `windowingBehavior`

Nécessité : Facultatif

Accepte : `"useNew"`, `"useExisting"`, `"useAnyExisting"`

Valeur par défaut : `"useNew"`

Taille du lancement

Colonnes au premier lancement

Il s'agit du nombre de colonnes de caractères affichées dans la fenêtre lors du premier chargement. Si `launchMode` a la valeur `"maximized"` ou `"maximizedFocus"`, cette propriété est ignorée.

Nom de la propriété : `initialCols`

Nécessité : Facultatif

Accepte : Entier

Valeur par défaut : `120`

Lignes au premier lancement

Il s'agit du nombre de lignes affichées dans la fenêtre lors du premier chargement. Si `launchMode` a la valeur `"maximized"` ou `"maximizedFocus"`, cette propriété est ignorée.

Nom de la propriété : `initialRows`

Nécessité : Facultatif

Accepte : Entier

Valeur par défaut : `30`

Position de lancement

Définit la position de pixel du coin supérieur gauche de la fenêtre lors du premier chargement. Sur un système avec plusieurs affichages, ces coordonnées sont fonction du coin supérieur gauche de l'affichage principal. Si aucune coordonnée X ou Y n'est fournie, le terminal utilise la valeur système par défaut pour cette valeur. Si `launchMode` a la valeur `"maximized"` ou `"maximizedFocus"`, la fenêtre est agrandie sur le moniteur spécifié par ces coordonnées.

Nom de la propriété : `initialPosition`

Nécessité : Facultatif

Accepte : Coordonnées sous forme de chaîne dans les formats suivants : `","`, `"#,#"`, `"#,"`, `",#"`

Valeur par défaut : `","`

Centrer au lancement

Quand la valeur est `true`, la fenêtre du terminal est automatiquement centrée à l'écran sur lequel elle s'ouvre. Le terminal utilise `"initialPosition"` pour déterminer l'écran à utiliser.

Les interactions avec les autres paramètres de lancement sont les suivantes :

- `"initialPosition": "x,y"`, `"centerOnLaunch": true`, `"launchMode": "default"` : centrer sur le moniteur localisé par `x,y`.
- `"initialPosition": "x,y"`, `"centerOnLaunch": true`, `"launchMode": "maximized"` : agrandir sur le moniteur localisé par `x,y` (`centerOnLaunch` n'ajoute rien).
- `"initialPosition": <omitted>`, `"centerOnLaunch": true`, `"launchMode": "default"` : centrer sur le moniteur par défaut.
- `"initialPosition": <omitted>`, `"centerOnLaunch": true`, `"launchMode": "focus"` : centrer et entrer le mode Focus sur le moniteur par défaut.
- `"initialPosition": <omitted>`, `"centerOnLaunch": true`, `"launchMode": "maximized"` : agrandir sur le moniteur par défaut (`centerOnLaunch` n'ajoute rien).

Nom de la propriété : `centerOnLaunch`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Désactiver les profils dynamiques

Permet de définir quels générateurs de profils dynamiques sont désactivés, ce qui les empêche d'ajouter leurs profils à la liste des profils lors du démarrage. Pour avoir des informations sur les profils dynamiques, consultez la page Profils dynamiques.

Nom de la propriété : `disabledProfileSources`

Nécessité : Facultatif

Accepte : `"Windows.Terminal.Wsl"`, `"Windows.Terminal.Azure"`, `"Windows.Terminal.PowershellCore"` et/ou `"Windows.Terminal.SSH"` à l'intérieur d'un tableau

Valeur par défaut : `[]`

Actions de démarrage

Vous pouvez définir une liste d'actions à exécuter au démarrage, ce qui permet de lancer le terminal avec un ensemble personnalisé d'onglets et de volets par défaut. Ces actions sont appliquées uniquement si aucun argument de ligne de commande n'est fourni. La liste d'actions est représentée par une chaîne au même format que les commandes dans les arguments de ligne de commande. Pour plus d'informations sur le format des commandes, consultez la page Arguments de ligne de commande.

Nom de la propriété : `startupActions`

Nécessité : Facultatif

Accepte : Chaîne représentant une liste de commandes à exécuter

Valeur par défaut : `""`

Paramètres d’interaction dans le Terminal Windows

Article • 05/10/2023

Les propriétés listées ci-dessous affectent tout le Terminal Windows, quels que soient les paramètres du profil. Elles doivent être placées à la racine de votre fichier settings.json.

Copier automatiquement la sélection dans le Presse-papiers

Quand cette valeur est définie sur `true`, une sélection est immédiatement copiée dans votre presse-papiers lors de la création. Dans ce cas, En cliquant sur le bouton droit de votre souris, vous effectuez toujours un collage. Lorsque la valeur est définie sur `false`, la sélection est conservée et attend une autre action. Appuyez sur le bouton droit de la souris pour copier la sélection.

Nom de la propriété : `copyOnSelect`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Format du texte lors de la copie

Quand la valeur est `true`, la mise en forme de la couleur et de la police du texte sélectionné est également copiée dans votre Presse-papiers. Lorsqu’elle est définie sur `false`, seul le texte brut est copié dans votre presse-papiers. Vous pouvez également spécifier les formats que vous souhaitez copier.

Nom de la propriété : `copyFormatting`

Nécessité : Facultatif

Accepte : `true`, `false`, `"all"`, `"none"`, `"html"`, `"rtf"`

Valeur par défaut : `false`

Supprimer l’espace blanc de fin dans la sélection rectangulaire

Lorsque la valeur est `true` et que vous copiez du texte d’une sélection rectangulaire (bloc) dans le Presse-papiers, les espaces blancs de fin sont supprimés de chaque ligne. Quand la valeur est `false`, les espaces blancs sont conservés, ce qui garantit que toutes les lignes ont la même longueur. Pour copier du texte dans une sélection rectangulaire (bloc), maintenez enfoncée la touche `Alt`, cliquez, puis faites glisser la souris sur la zone de texte à sélectionner. Cela peut être utile pour sélectionner des colonnes de texte, etc.

Nom de la propriété : `trimBlockSelection`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`

Couper les espaces blancs de fin sur la pâte

Quand cette option est activée, le terminal supprime automatiquement les espaces de fin lors du collage de texte dans le terminal.

Nom de la propriété : `trimPaste`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`

Séparateurs de mots

Détermine les séparateurs de mots utilisés dans une sélection par double-clic. Les séparateurs de mots sont des caractères qui spécifient la limite entre deux mots. Les exemples les plus courants sont les espaces, les points-virgules, les virgules et les points.

Nom de la propriété : `wordDelimiters`

Nécessité : Facultatif

Accepte : Caractères sous forme de chaîne

Valeur par défaut : `/\\()\"'-:,.;<>~!@#$%^&*|+=[]{}?|`

(`|` is `U+2502 BOX DRAWINGS LIGHT VERTICAL`)

> **ⓘ Important**
>
> Les caractères suivants doivent être placés dans une séquence d'échappement avec une barre oblique inverse : `\`, `"`

Aligner le redimensionnement de la fenêtre sur la grille de caractères

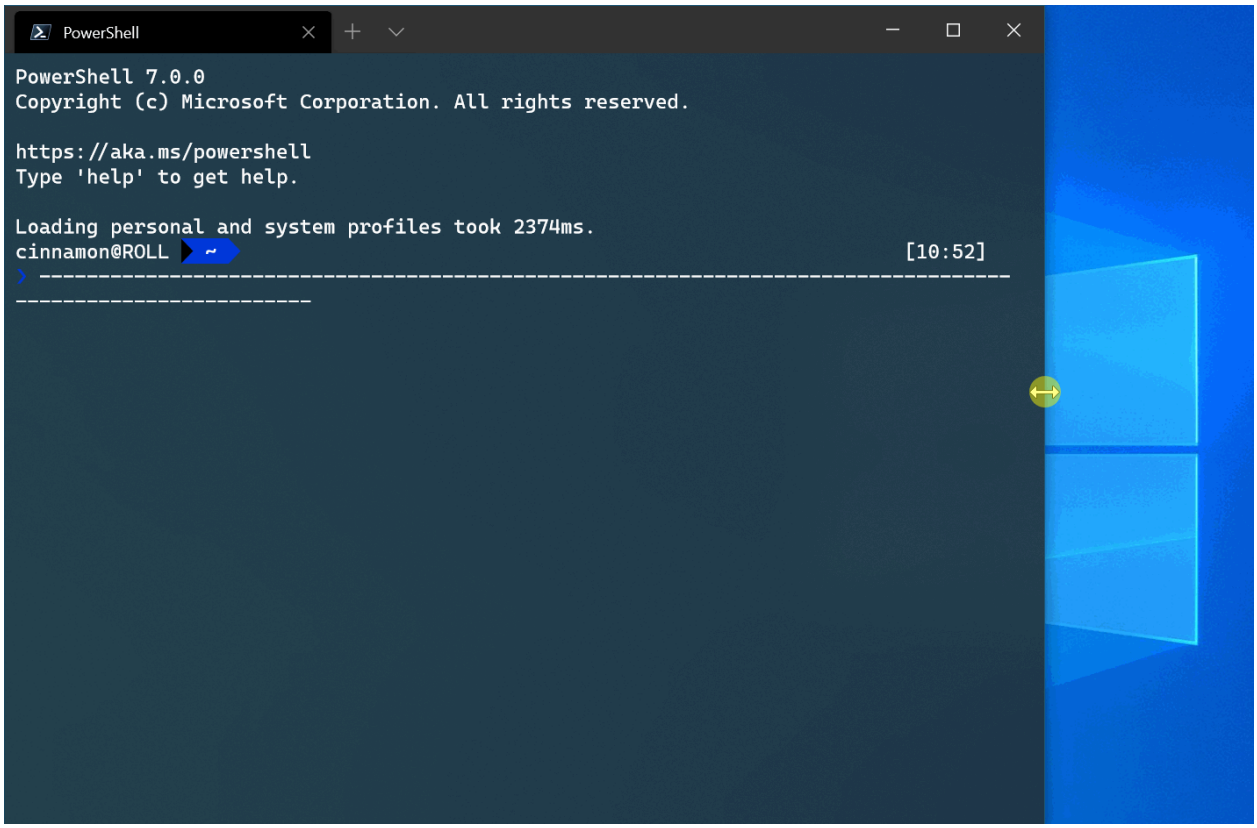
Quand cette valeur est définie sur `true`, la fenêtre s'aligne sur la limite de caractère la plus proche lors du redimensionnement. Lorsqu'elle est définie sur `false`, la fenêtre se redimensionne progressivement.

Nom de la propriété : `snapToGridOnResize`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`



Réduire à la zone de notification

Quand la valeur est `true`, la réduction d'une fenêtre entraîne sa disparition de la barre des tâches, ce qui la rend inaccessible à partir de cette zone. Vous pouvez par contre y accéder à l'aide de l'icône de zone de notification du terminal. Si ce paramètre global ou le paramètre global `minimizeToNotificationArea` est true, le terminal place une icône dans la zone de notification.

Nom de la propriété : `minimizeToNotificationArea`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

> ⓘ **Important**
>
> Ce paramètre, anciennement `"minimizeToTray"`, s'appelle désormais `"minimizeToNotificationArea"`.

Toujours afficher l'icône notification

Quand la valeur est `true`, le terminal place son icône dans la zone de notification. Si ce paramètre global ou le paramètre global `minimizeToNotificationArea` est true, le terminal place une icône dans la zone de notification. L'utilisateur peut également utiliser l'*action* `minimizeToNotificationArea`.

Nom de la propriété : `alwaysShowNotificationIcon`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

> ⓘ **Important**
>
> Ce paramètre, anciennement `"alwaysShowTrayIcon"`, s'appelle désormais `"alwaysShowNotificationIcon"`.

Paramètres des onglets

Style d’interface du sélecteur d’onglet

Quand la valeur est `true` ou `"mru"`, les commandes `nextTab` et `prevTab` utilisent l’interface utilisateur du sélecteur d’onglet, avec le dernier classement utilisé. Quand la valeur est `"inOrder"`, ces actions permettent de passer à un autre onglet selon l’ordre actuel des onglets dans la barre d’onglets. L’interface utilisateur affiche tous les onglets actuellement ouverts dans une liste verticale que vous pouvez parcourir avec le clavier ou la souris.

Le sélecteur d’onglet s’ouvre dès l’activation initiale des actions `nextTab` et `prevTab`, et reste ouvert tant qu’une touche de modification est maintenue enfoncée. Quand toutes les touches de modification sont relâchées, le sélecteur se ferme et l’onglet en surbrillance a le focus. Vous pouvez utiliser `Tab`/`Maj+Tab`, les touches de direction `haut` et `bas` ainsi que les actions `nextTab`/`prevTab` pour parcourir l’interface utilisateur du sélecteur.

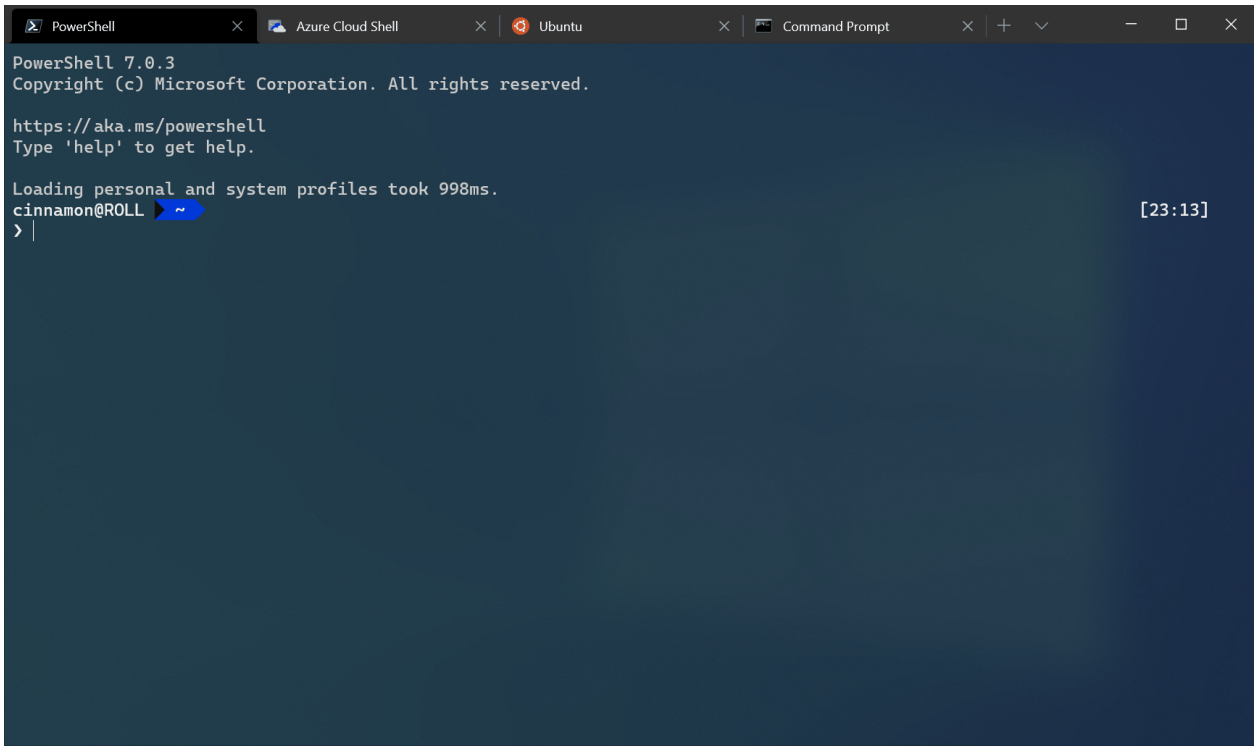
Pour désactiver le sélecteur d’onglet, utilisez la valeur `false` ou `"disabled"`.

Nom de la propriété : `tabSwitcherMode`

Nécessité : Facultatif

Accepte : `true`, `false`, `"mru"`, `"inOrder"`, `"disabled"`

Valeur par défaut : `"inOrder"`



Activer le sélecteur d'onglet

Quand la valeur est `true`, les commandes `nextTab` et `prevTab` utilisent l'interface utilisateur du sélecteur d'onglet. L'interface utilisateur affiche tous les onglets actuellement ouverts dans une liste verticale que vous pouvez parcourir avec le clavier ou la souris.

Le sélecteur d'onglet s'ouvre dès l'activation initiale des actions `nextTab` et `prevTab`, et reste ouvert tant qu'une touche de modification est maintenue enfoncée. Quand toutes les touches de modification sont relâchées, le sélecteur se ferme et l'onglet en surbrillance a le focus. Vous pouvez utiliser `Tab`/`Maj+Tab`, les touches de direction `haut` et `bas` ainsi que les actions `nextTab`/`prevTab` pour parcourir l'interface utilisateur du sélecteur.

Nom de la propriété : `useTabSwitcher`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`

> ⊗ **Attention**

Le paramètre `"useTabSwitcher"` n'est plus disponible dans les versions 1,5 et ultérieures. Nous vous recommandons d'utiliser le paramètre `"tabSwitcherMode"` à la place.

Masquer automatiquement en cas de perte de focus

Quand cette option est activée, la fenêtre du terminal est automatiquement masquée dès que la fenêtre perd le focus.

Nom de la propriété : `autoHideWindow`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

—

Déplacer automatiquement le focus sur le volet pointé par la souris

Quand la valeur est `true`, le terminal déplace le focus sur le volet pointé par la souris. Quand la valeur est `false`, un clic est nécessaire pour déplacer le focus sur le volet avec la souris.

Nom de la propriété : `focusFollowMouse`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Détecter automatiquement les URL et les rendre cliquables

Quand la valeur est `true`, les URL sont détectées par le terminal. Les URL sont alors soulignées quand vous pointez dessus. Vous pouvez également cliquer sur celles-ci en appuyant sur `Ctrl`. Il s'agit d'une fonctionnalité expérimentale dont l'existence à long terme n'est pas garantie.

Nom de la propriété : `experimental.detectURLs`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`

Avertissements relatifs au collage

Avertir quand le texte à coller est très grand

Quand la valeur est `true`, la tentative de collage d'un texte contenant plus de 5 Kio de caractères entraîne l'ouverture d'une boîte de dialogue vous demandant si vous souhaitez continuer ou non. Quand la valeur est `false`, la boîte de dialogue ne s'affiche pas et le texte est collé immédiatement. S'il vous arrive souvent de cliquer accidentellement avec le bouton droit de la souris sur le terminal après avoir sélectionné beaucoup de texte, cette propriété peut être utile pour empêcher le terminal de se bloquer quand le programme connecté au terminal reçoit le contenu du Presse-papiers.

Nom de la propriété : `largePasteWarning`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`

Avertir quand le texte à coller contient plusieurs lignes

Quand la valeur est `true`, la tentative de collage d'un texte contenant plusieurs lignes entraîne l'ouverture d'une boîte de dialogue vous demandant si vous souhaitez continuer ou non. Quand la valeur est `false`, la boîte de dialogue ne s'affiche pas et le texte est collé immédiatement. Dans la plupart des shells, une ligne correspond à une commande. Par conséquent, si vous collez du texte qui contient le caractère « nouvelle ligne » dans un shell, une ou plusieurs commandes peuvent être exécutées

automatiquement après l'opération de collage, sans que vous ayez le temps de les valider. Cette propriété peut être utile si vous copiez et collez souvent des commandes provenant de sites web non approuvés.

Nom de la propriété : `multiLinePasteWarning`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`

Encodage d'entrée hérité

Force le terminal à utiliser l'encodage d'entrée hérité. Des touches spécifiques dans certaines applications peuvent cesser de fonctionner lorsque ce paramètre est activé, mais il peut s'avérer utile dans des scénarios avancés de débogage de problèmes d'entrée, en particulier en mode « debug tap ».

Nom de la propriété : `experimental.input.forceVT`

Nécessité : Facultatif

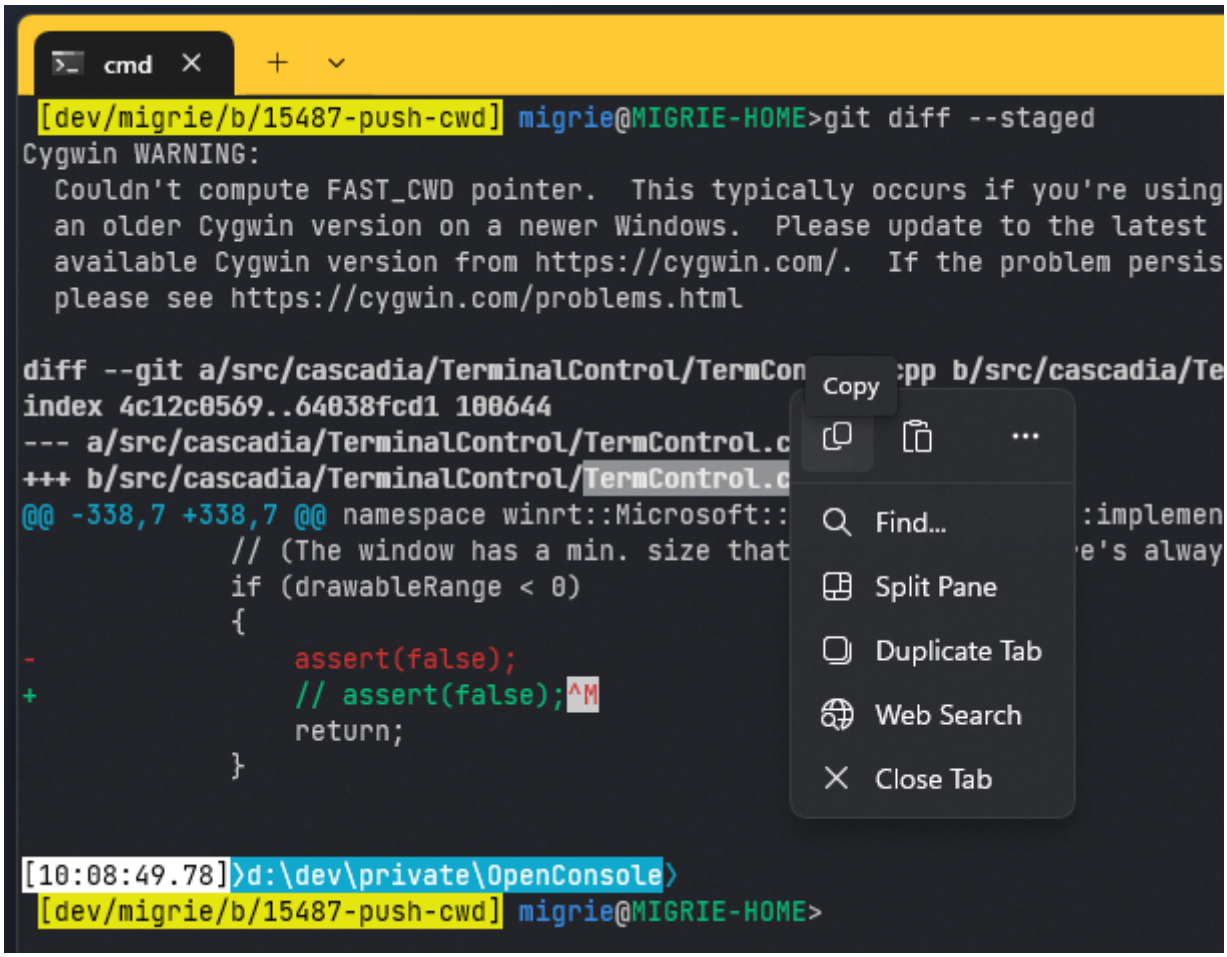
Accepte : `true`, `false`

Valeur par défaut : `false`

Menu contextuel

Le menu contextuel du Terminal Windows est un moyen simple d'accéder rapidement aux actions courantes. Quand la valeur est `true`, un clic droit dans le Terminal active le menu contextuel. Quand la valeur est `false`, un clic droit colle le texte.

Le menu contextuel peut également être ouvert avec l'action `showContextMenu`, que ce paramètre soit activé ou non.



Nom de la propriété : `experimental.rightClickContextMenu`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Recherche sur le web

Il s'agit de l'URL par défaut utilisée pour la recherche sur le web à partir du terminal avec l'action `searchWeb` (y compris le menu contextuel accessible par bouton droit). Dans cette chaîne, `%s` est remplacé par le texte sélectionné. La valeur par défaut est `https://www.bing.com/search?q=%s`.

Nom de la propriété : `searchWebDefaultQueryUrl`

Nécessité : Facultatif

Accepte : URL sous forme de chaîne

Valeur par défaut : `https://www.bing.com/search?q=%s`

> **ⓘ Important**
>
> Cette fonctionnalité n'est disponible que dans la **préversion de Terminal Windows**.

Paramètres d'apparence dans le Terminal Windows

Article • 04/10/2023

Les propriétés listées ci-dessous affectent tout le Terminal Windows, quels que soient les paramètres du profil. Elles doivent être placées à la racine de votre fichier settings.json.

Langage

Définit un remplacement pour la langue par défaut de l'application.

Nom de la propriété : `language`

Nécessité : Facultatif

Accepte : Balise de langue BCP-47 comme `"en-US"`

Thème

Cela définit le thème (thème sombre ou clair) de l'application. `"system"` utilise le même thème que Windows.

Nom de la propriété : `theme`

Nécessité : Facultatif

Accepte : `"system"`, `"dark"`, `"light"`, nom du thème personnalisé

Valeur par défaut : `"system"`

Toujours afficher les onglets

Lorsque cette valeur est définie sur `true`, les onglets sont toujours affichés. Lorsque cette valeur est définie sur `false` et que `showTabsInTitlebar` est défini sur `false`, les onglets sont toujours affichés sous la barre de titre. Lorsque ce paramètre est `false` et que `showTabsInTitlebar` est `false`, les onglets s'affichent uniquement s'il y en a plusieurs, en tapant `Ctrl + Maj + t` ou la combinaison de touches affectée à `newTab`.

Notez que la modification de ce paramètre nécessite le démarrage d'une nouvelle instance du terminal.

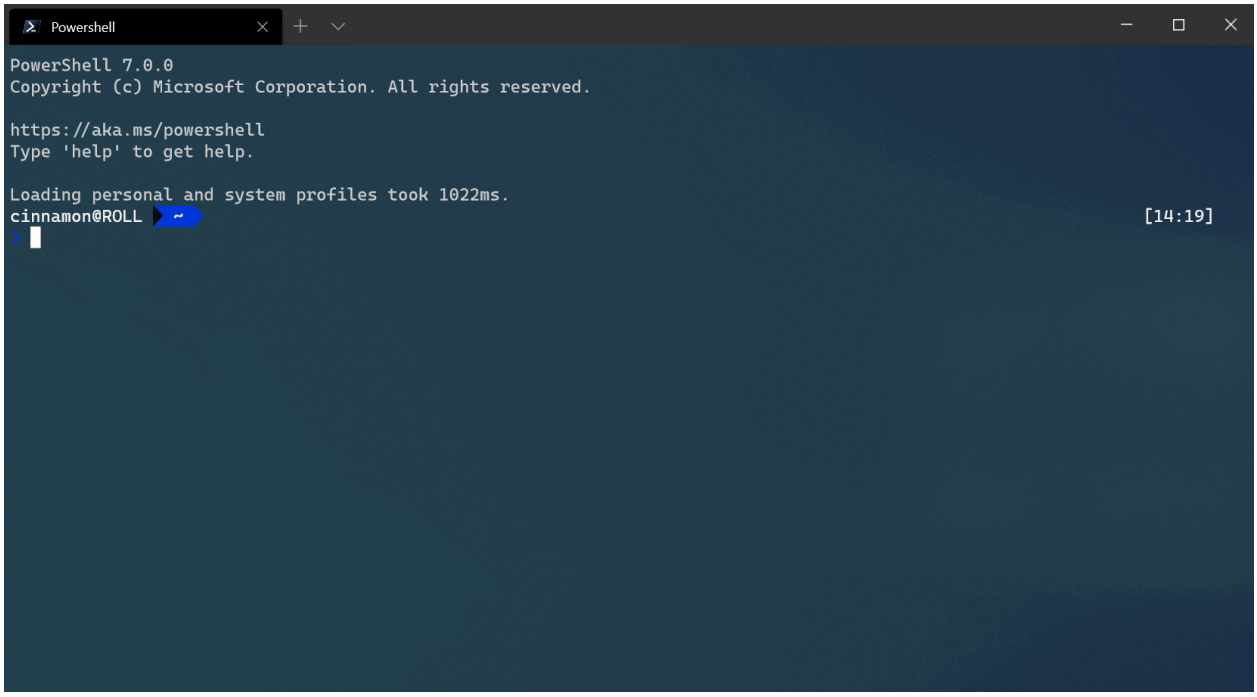
> ⓘ Notes
>
> Ce paramètre n'a aucun effet quand `showTabsInTitlebar` est `true`.

Nom de la propriété : `alwaysShowTabs`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`



Position des onglets nouvellement créés (préversion)

Spécifie l'emplacement où les nouveaux onglets s'affichent dans la ligne d'onglets. Lorsque cette valeur est définie sur `"afterLastTab"`, les nouveaux onglets s'affichent à la fin de la ligne d'onglets. Lorsque cette valeur est définie sur `"afterCurrentTab"`, les nouveaux onglets s'affichent après l'onglet actif.

Nom de la propriété : `newTabPosition`

Nécessité : Facultatif

Accepte : `"afterLastTab"`, `"afterCurrentTab"`

Valeur par défaut : `"afterLastTab"`

Masquer la barre de titre

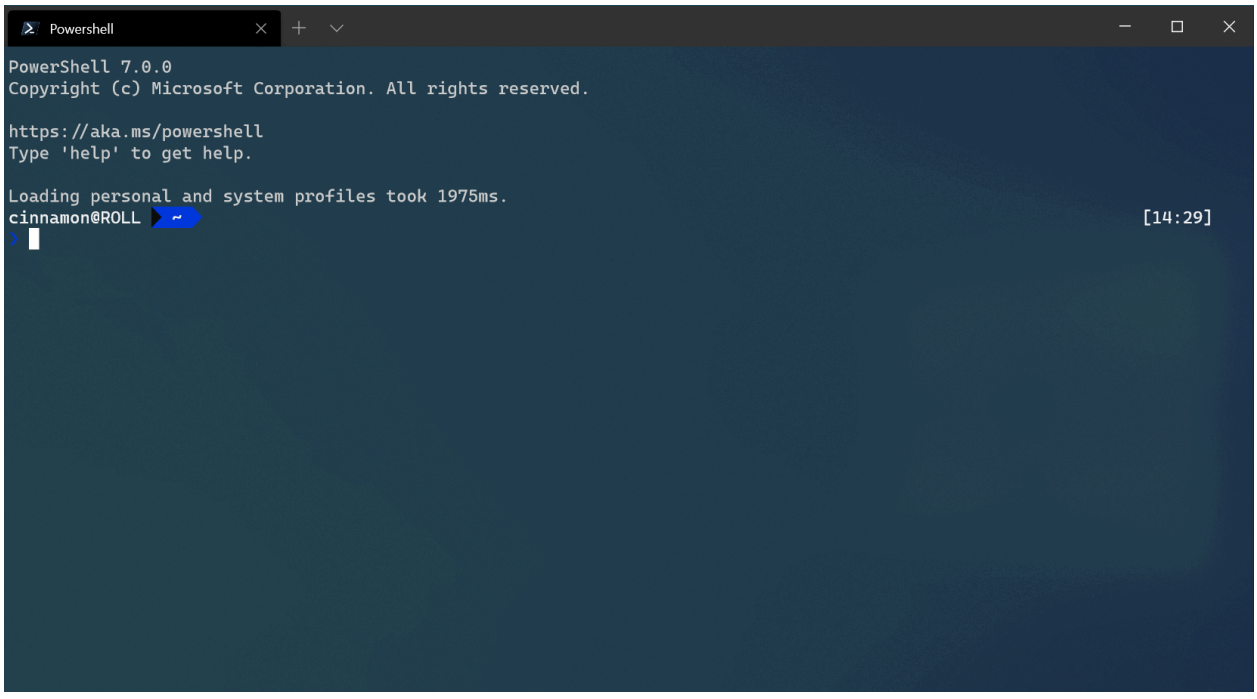
Lorsque cette valeur est définie sur `true`, les onglets sont déplacés dans la barre de titre et la barre de titre disparaît. Lorsqu'elle est définie sur `false`, la barre de titre se trouve au-dessus des onglets. Notez que la modification de ce paramètre nécessite le démarrage d'une nouvelle instance du terminal.

Nom de la propriété : `showTabsInTitlebar`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`



Afficher l'acrylique dans la ligne d'onglets

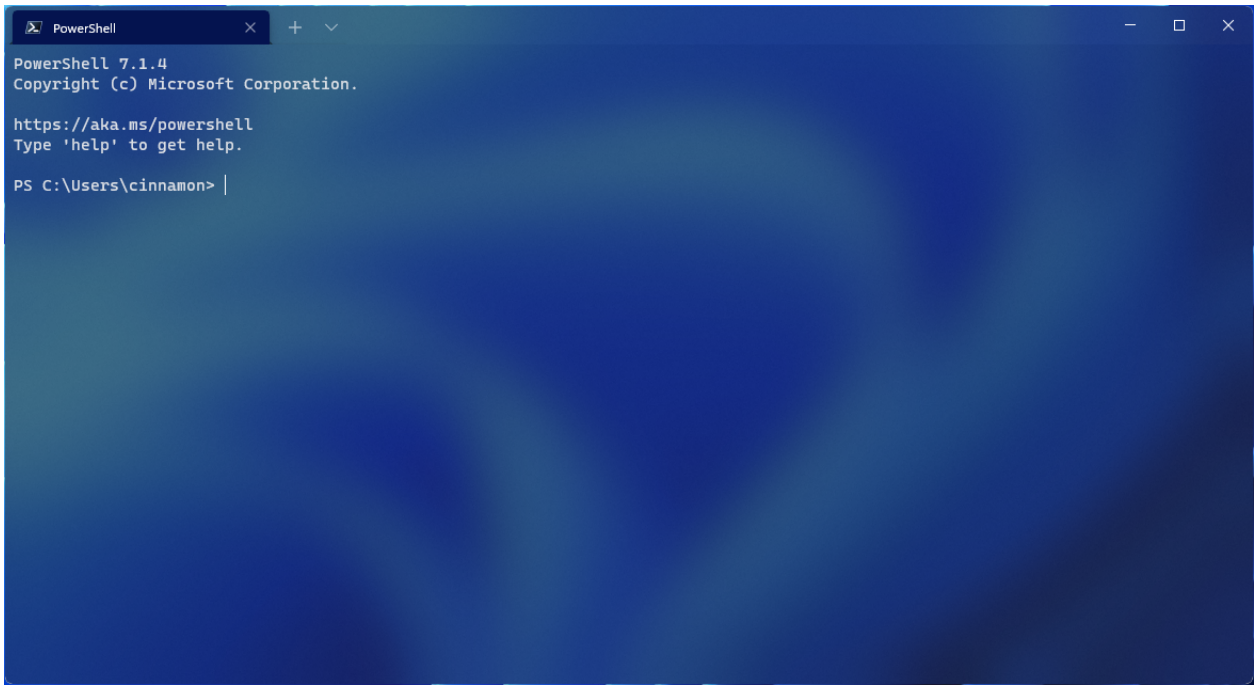
Quand la valeur est `true`, la ligne d'onglets reçoit un arrière-plan acrylique à 50 % d'opacité. Quand la valeur est `false`, la ligne d'onglets est opaque. Notez que la modification de ce paramètre nécessite le démarrage d'une nouvelle instance du terminal.

Nom de la propriété : `useAcrylicInTabRow`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`



Utiliser le titre du terminal actif comme titre de l'application

Quand cette valeur est définie sur `true`, la barre de titre affiche le titre de l'onglet sélectionné. Quand la valeur est `false`, la barre de titre affiche « Terminal Windows ». Notez que la modification de ce paramètre nécessite le démarrage d'une nouvelle instance du terminal.

Nom de la propriété : `showTerminalTitleInTitlebar`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`

Mode Toujours visible

Quand la valeur est true, la fenêtre du Terminal Windows est placée devant toutes les autres fenêtres du bureau. Vous pouvez également activer ou désactiver cet état avec la combinaison de touches `toggleAlwaysOnTop`.

Nom de la propriété : `alwaysOnTop`

Nécessité : Facultatif

Accepte : `true, false`

Valeur par défaut : `false`

Mode de largeur des onglets

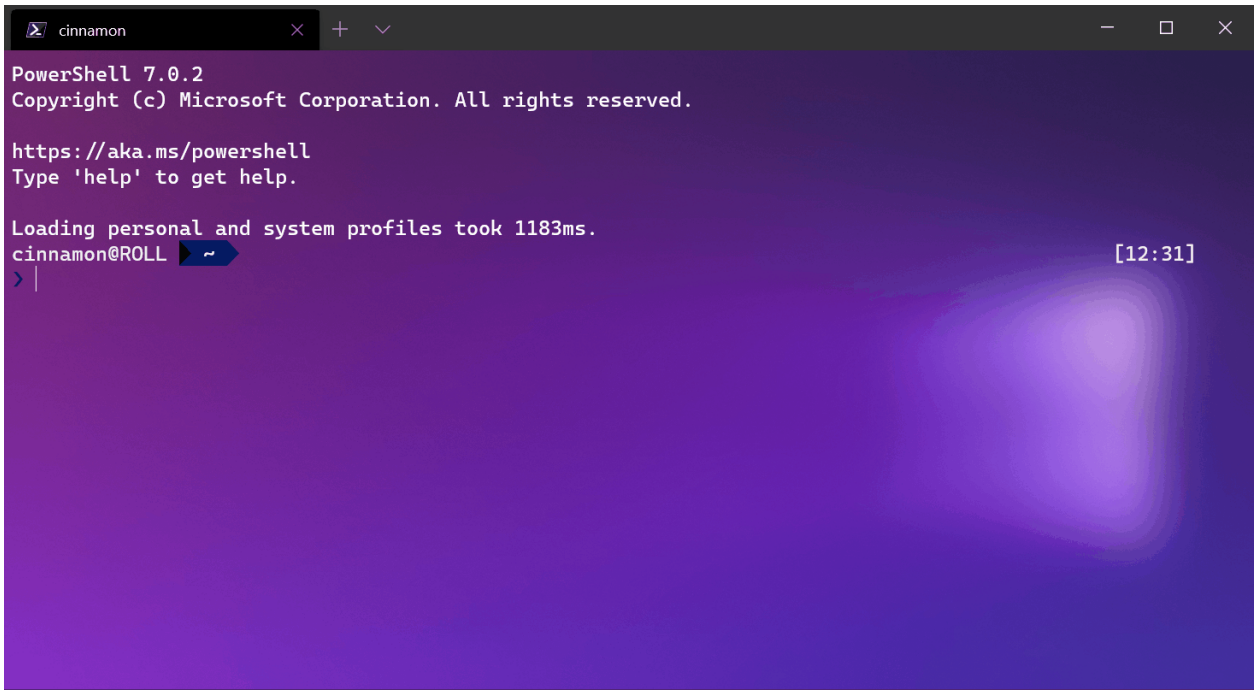
Ceci permet de définir la largeur des onglets. `"equal"` donne la même largeur à chaque onglet. `"titleLength"` dimensionne chaque onglet en fonction de la longueur de son titre. `"compact"` réduit tous les onglets inactifs à la largeur de l'icône, en laissant ainsi plus d'espace à l'onglet actif pour afficher son titre complet.

Nom de la propriété : `tabWidthMode`

Nécessité : Facultatif

Accepte : `"equal"`, `"titleLength"`, `"compact"`

Valeur par défaut : `"equal"`



Désactiver les animations du volet

Quand la valeur est `true`, désactive les animations visuelles dans toute l'application.

Nom de la propriété : `disableAnimations`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Afficher la fenêtre contextuelle Fermer tous les onglets

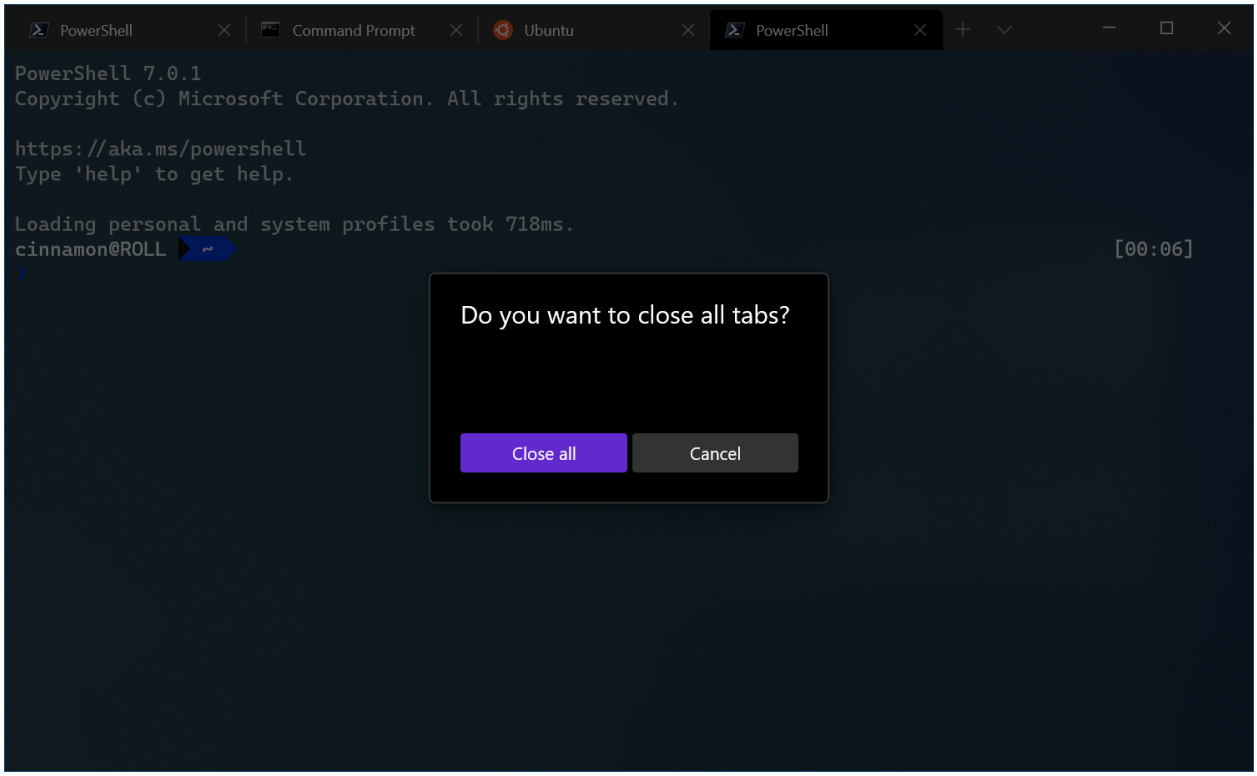
Lorsque cette valeur est définie sur `true`, la fermeture d'une fenêtre où plusieurs onglets sont ouverts *nécessite* une confirmation. Lorsque cette valeur est définie sur `false`, la fermeture d'une fenêtre où plusieurs onglets sont ouverts *ne nécessite pas* de confirmation.

Nom de la propriété : `confirmCloseAllTabs`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`



Utiliser une image de fond pour toute la fenêtre

Lorsqu'il est défini sur `true`, l'image d'arrière-plan du profil actuellement ciblé est agrandie pour englober toute la fenêtre, sous les autres volets. Il s'agit d'une fonctionnalité expérimentale et sa pérennité n'est pas garantie.

Nom de la propriété : `experimental.useBackgroundImageForWindow`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Liste déroulante du nouvel onglet

Ce paramètre vous permet de configurer la liste des profils et la structure du menu déroulant du nouvel onglet. Cela vous permet de réorganiser les profils, d'imbriquer les

profils dans des sous-menus, de masquer les profils et bien plus encore. Le paramètre `newTabMenu` accepte une liste d'« entrées de menu Nouvel onglet », décrites ci-dessous.

Un exemple de ce paramètre pourrait ressembler à :

JSON

```
{
    "newTabMenu": [
        { "type":"profile", "profile": "Command Prompt" },
        { "type":"profile", "profile": "Windows PowerShell" },
        { "type":"separator" },
        {
            "type":"folder",
            "name": "ssh",
            "icon": "C:\\path\\to\\icon.png",
            "entries":
            [
                { "type":"profile", "profile": "Host 1" },
                { "type":"profile", "profile": "8.8.8.8" },
                { "type":"profile", "profile": "Host 2" }
            ]
        },
        {
            "type": "folder",
            "name": "WSL",
            "entries": [ { "type": "matchProfile", "source":
"Microsoft.Terminal.Wsl" } ]
        },
        { "type": "remainingProfiles" }
    ]
}
```

Nom de la propriété : `newTabMenu`

Nécessité : Facultatif

Accepte : une liste des nouvelles entrées du menu de l'onglet

Valeur par défaut : `[ { "type":"remainingProfiles" } ]`

Entrées du menu des nouveaux onglets

Voici différents types de nouvelles entrées de menu d'onglets qui peuvent être utilisées dans le paramètre `newTabMenu`. Ils se présentent chacun sous la forme d'un objet JSON avec une propriété `type` et d'autres propriétés spécifiques à ce type d'entrée. Les valeurs de la propriété `type` sont répertoriées ci-dessous.

- profile
- folder
- separator
- matchProfile
- remainingProfiles

Profil

Ce type d'entrée représente un profil de votre liste de profils. Le profil peut être spécifié par nom ou GUID.

JSON

```
{ "type":"profile", "profile": "Command Prompt" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`profile`	Obligatoire	Nom ou GUID du profil sous forme de chaîne	Le profil qui s'ouvre en fonction de son GUID ou de son nom.

Folder

Ce type d'entrée représente un dossier imbriqué dans le menu déroulant du nouvel onglet. Les dossiers peuvent être imbriqués dans d'autres dossiers.

JSON

```
{
    "type":"folder",
    "name": "ssh",
    "icon": "C:\\path\\to\\icon.png",
    "entries":
    [
        { "type":"profile", "profile": "Host 1" },
        { "type":"profile", "profile": "Host 2" }
    ]
}
```

Paramètres

Nom	Nécessité	Accepte	Description
`name`	Obligatoire	Nom du dossier sous forme de chaîne	Nom du dossier, affiché sur l'entrée de menu.
`icon`	Facultatif	Chemin d'accès à une icône sous forme de chaîne	Chemin d'accès à une icône qui sera affichée à côté du nom du dossier.
`entries`	Requis	Liste des entrées du menu des nouveaux onglets	Liste des nouvelles entrées de menu d'onglets qui seront affichées lorsque vous cliquerez sur le dossier.
`allowEmpty`	Facultatif	Booléen (par défaut : `true`)	S'il est défini sur `true`, le dossier sera affiché même s'il ne contient aucune entrée. S'il est défini sur `false`, le dossier ne sera pas affiché s'il ne contient aucune entrée. Cela peut être utile avec les entrées `matchProfile`.
`inline`	Facultatif	Booléen (par défaut : `false`)	S'il est défini sur `true` et qu'il n'y a qu'une seule entrée dans le dossier, ce dossier ne créera pas de menu imbriqué. Au lieu de cela, l'entrée dans le menu ne sera que la seule entrée du dossier. Cela peut être utile avec les entrées `matchProfile`.

Séparateur

Ce type d'entrée représente un séparateur dans le menu déroulant du nouvel onglet.

JSON

```
{ "type":"separator" }
```

Profils restants

Ce type d'entrée représente tous les profils qui ne sont pas déjà représentés dans le menu déroulant du nouvel onglet. Ceci est utile si vous souhaitez qu'un ensemble de profils soient toujours affichés en haut du menu déroulant du nouvel onglet, puis que le reste des profils soit affiché dans un dossier en bas du menu déroulant du nouvel onglet.

Cela renverra une liste des profils restants, dans l'ordre dans lequel ils apparaissent dans la liste `profiles`.

```JSON
{ "type": "remainingProfiles" }
```

Profil de correspondance

Ce type d'entrée est similaire à l'entrée de profils restante. Cette entrée s'étendra à une liste de profils qui correspondent tous à une propriété donnée. Vous pouvez faire une correspondance en fonction des profils par `name`, `commandline`, ou `source`.

Par exemple :

```JSON
{ "type": "matchProfile", "source": "Microsoft.Terminal.Wsl" }
```

Créera un ensemble d'entrées qui sont tous des profils avec la propriété `source` définie sur `Microsoft.Terminal.Wsl`. Une comparaison de chaîne complète est effectuée sur ces propriétés – pas une correspondance d'expression régulière ou de chaîne partielle.

Paramètres

Nom	Nécessité	Accepte	Description
`name`	Facultatif	Nom du profil sous forme de chaîne	Une valeur à comparer à celle `name` du profil.
`commandline`	Facultatif	Ligne de commande sous forme de chaîne	Une valeur à comparer à celle `commandline` du profil.
`source`	Facultatif	Source du profil sous forme de chaîne	Une valeur à comparer à celle `source` du profil.

Jeux de couleurs dans Terminal Windows

Article • 21/03/2023

Le Terminal Windows vous permet de définir vos propres modèles de couleurs. Pour cela, vous pouvez utiliser les modèles prédéfinis intégrés ou créer entièrement votre propre modèle. Pour modifier des modèles, vous devez modifier le fichier settings.json dans un éditeur tel que Visual Studio Code.

Utilisation d'un autre modèle de couleurs

Lancez le Terminal Windows, puis sélectionnez la petite flèche allant vers le bas dans la barre de titre. Dans le menu déroulant qui s'ouvre, vous pouvez voir les profils disponibles sur votre système (par exemple, Windows PowerShell et Invite de commandes) ainsi que d'autres options. Sélectionnez **Paramètres**. Le fichier settings.json s'ouvre dans votre éditeur de texte par défaut.

Ce fichier vous permet de définir différentes options par fenêtre ou par profil. Pour illustrer cela, nous allons changer le modèle de couleurs du profil Invite de commandes.

Parcourez le fichier JSON jusqu'à la section comprenant :

JSON

```
"commandline": "cmd.exe",
"hidden": false
```

Remplacez-la par ce qui suit :

JSON

```
"commandline": "cmd.exe",
"hidden": false,
"colorScheme": "Tango Light"
```

Notez la virgule supplémentaire dans la ligne **hidden**. Une fois ce fichier enregistré, le Terminal Windows met à jour toutes les fenêtres ouvertes. Ouvrez un onglet Invite de commandes si ce n'est pas déjà fait, et vous verrez immédiatement que les couleurs ont été modifiées.

Création de votre propre modèle de couleurs

Le modèle « Tango Light » est inclus comme option par défaut, mais vous pouvez créer votre propre modèle (en copiant un modèle existant ou en partant de rien).

Les modèles de couleurs peuvent être définis dans le tableau `schemes` de votre fichier settings.json. Ils sont écrits dans le format suivant :

JSON

```
{
    "name" : "Campbell",

    "cursorColor": "#FFFFFF",
    "selectionBackground": "#FFFFFF",

    "background" : "#0C0C0C",
    "foreground" : "#CCCCCC",

    "black" : "#0C0C0C",
    "blue" : "#0037DA",
    "cyan" : "#3A96DD",
    "green" : "#13A10E",
    "purple" : "#881798",
    "red" : "#C50F1F",
    "white" : "#CCCCCC",
    "yellow" : "#C19C00",
    "brightBlack" : "#767676",
    "brightBlue" : "#3B78FF",
    "brightCyan" : "#61D6D6",
    "brightGreen" : "#16C60C",
    "brightPurple" : "#B4009E",
    "brightRed" : "#E74856",
    "brightWhite" : "#F2F2F2",
    "brightYellow" : "#F9F1A5"
},
```

Chaque paramètre, en dehors de `name`, accepte une couleur sous forme de chaîne au format hexadécimal : `"#rgb"` ou `"#rrggbb"`. Les paramètres `cursorColor` et `selectionBackground` sont facultatifs.

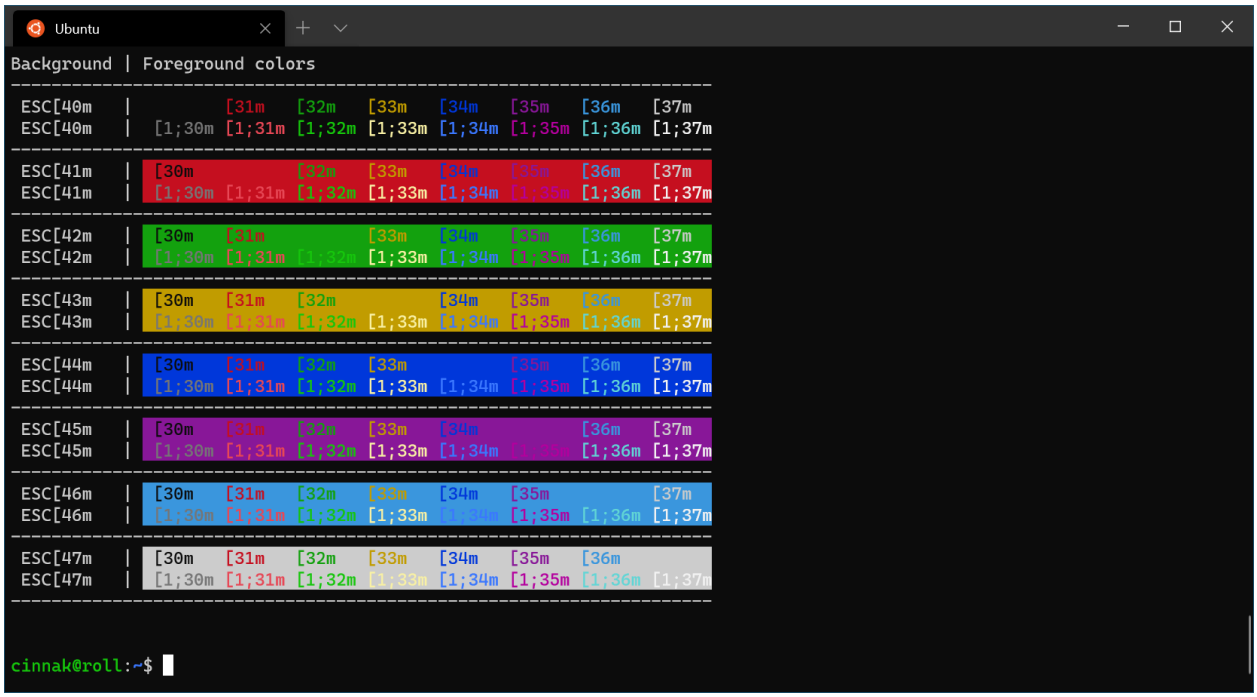
Modèles de couleurs inclus

Windows Terminal intègre ces jeux de couleurs dans le fichier defaults.json, accessible en maintenant la touche `alt` appuyée et en sélectionnant le bouton Paramètres. Les modèles de couleurs **ne peuvent pas** être modifiés dans le fichier defaults.json. Pour

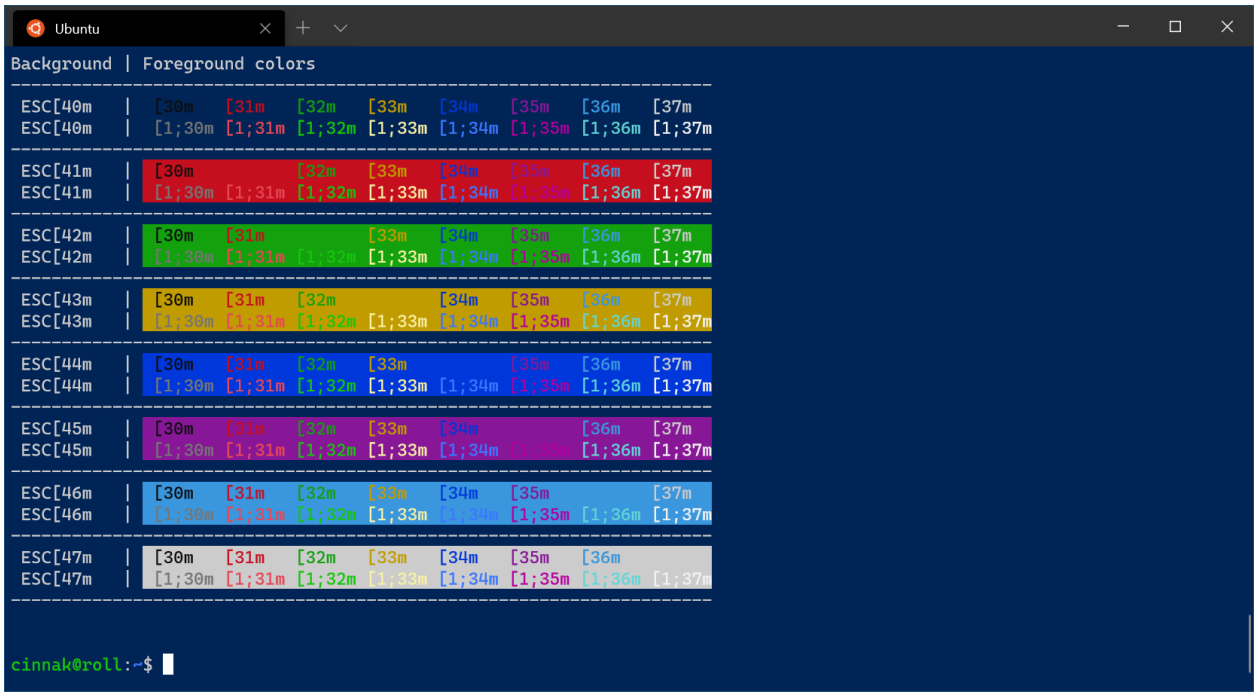
qu'un modèle de couleurs s'applique à tous les profils, modifiez-le dans la section defaults de votre fichier settings.json.

> ⓘ Notes
>
> Vous pouvez afficher le modèle de couleurs actuel sur le Terminal en utilisant **colortool**, avec la ligne de commande `colortool -c`

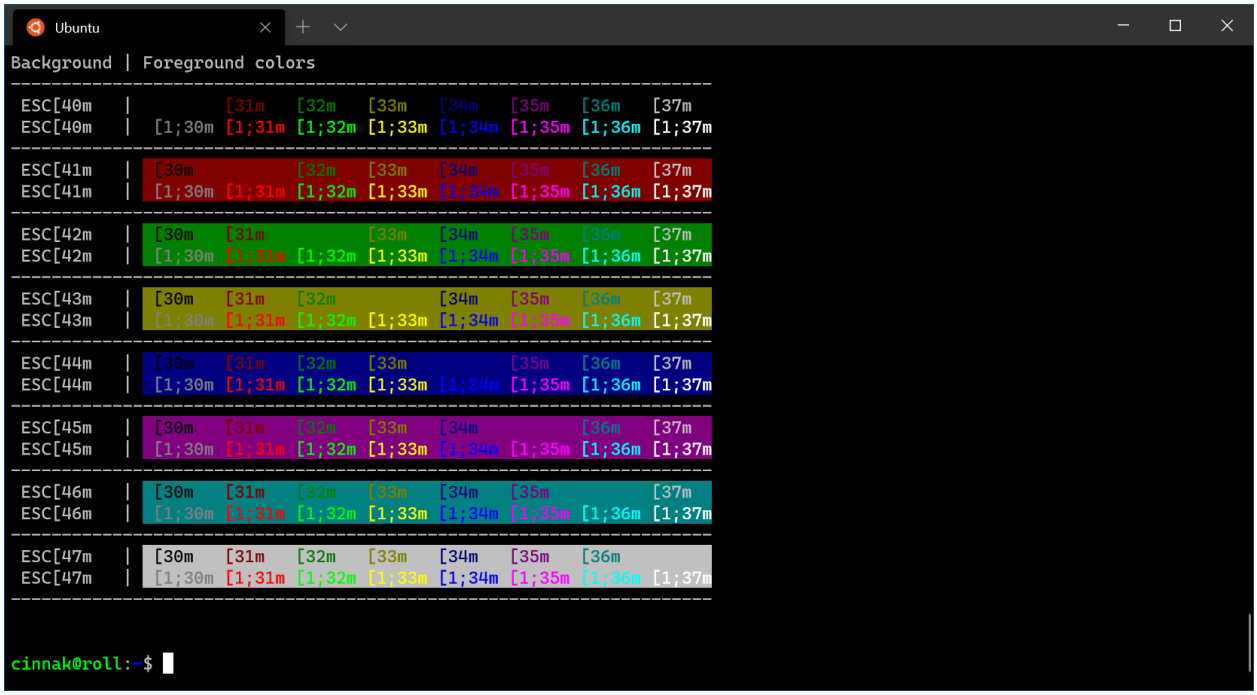
Campbell



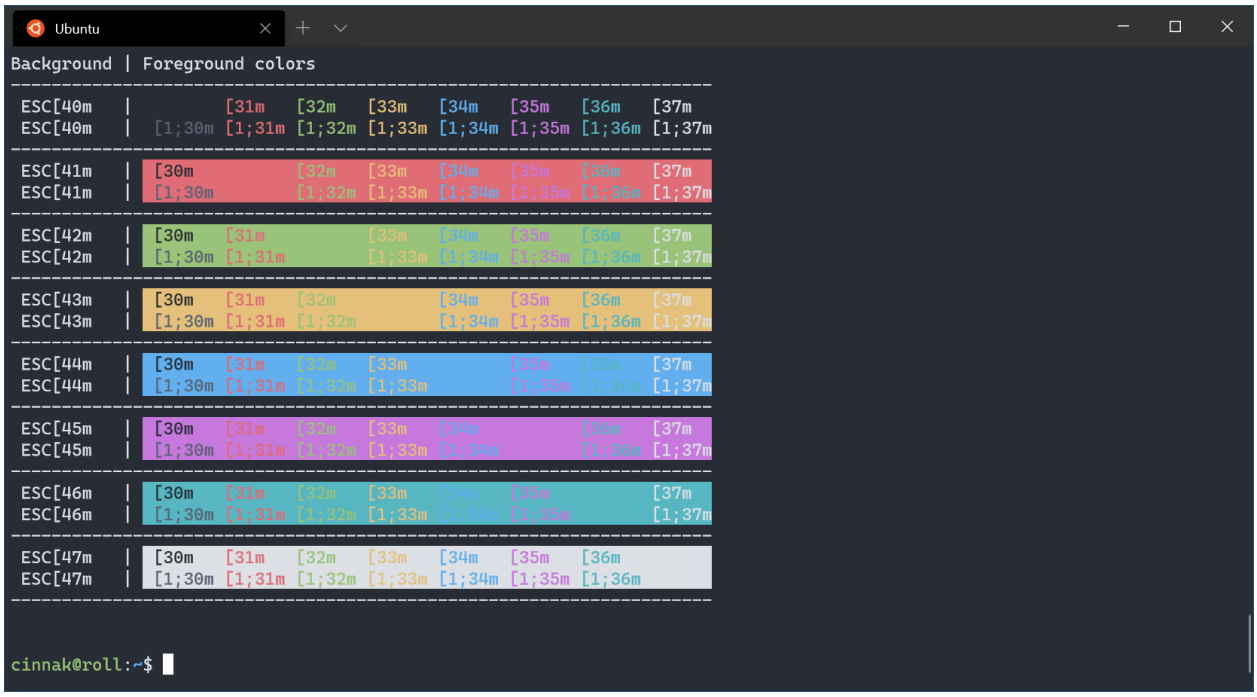
Campbell Powershell



Vintage



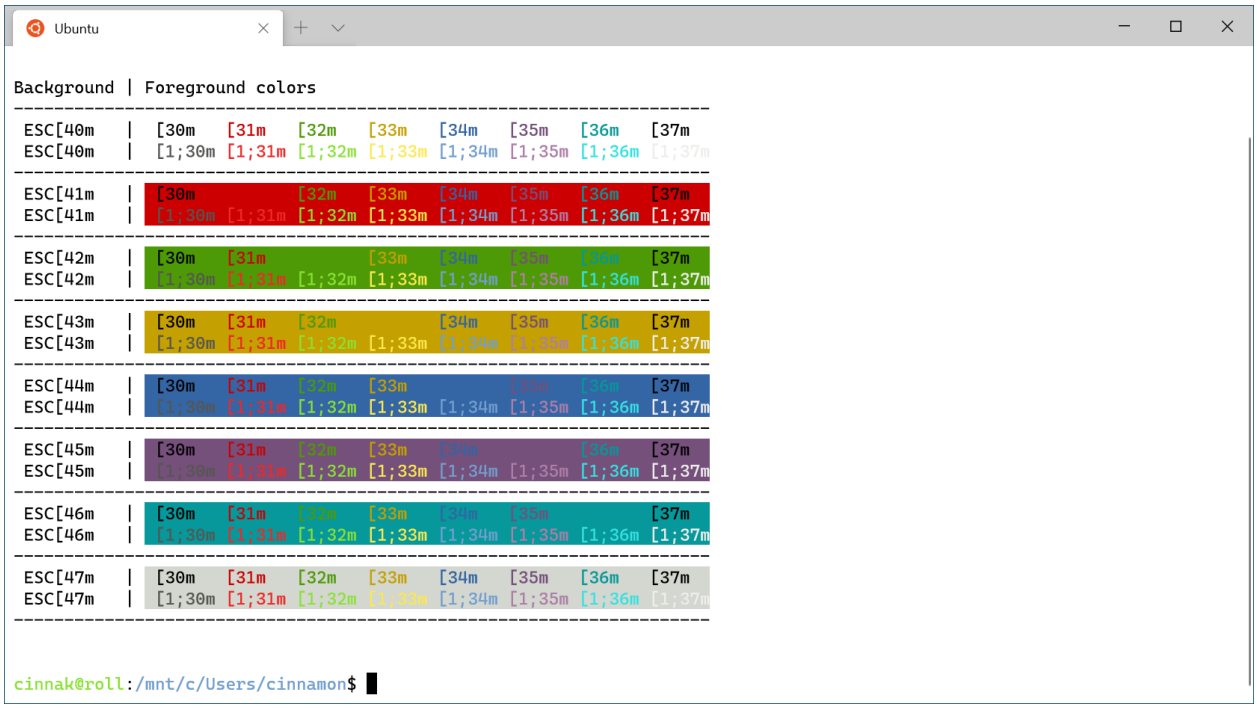
One Half Dark



One Half Light

Tango Dark

Tango Light



Autres modèles

Pour plus de modèles, consultez la section Galerie de terminaux personnalisés.

Paramètres de rendu dans le Terminal Windows

Article • 04/10/2023

Les propriétés listées ci-dessous affectent tout le Terminal Windows, quels que soient les paramètres du profil. Elles doivent être placées à la racine de votre fichier settings.json.

Si vous envisagez de modifier les paramètres de rendu, des informations supplémentaires sont fournies dans la page de résolution des problèmes pour vous guider.

Redessiner l'écran entier lors de l'affichage de mises à jour

Quand la valeur est `true`, le terminal redessine la totalité de l'écran pour chaque image. Quand la valeur est `false`, il affiche uniquement les mises à jour de l'écran entre les images.

Nom de la propriété : `experimental.rendering.forceFullRepaint`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Utiliser le rendu logiciel

Quand la valeur est `true`, le terminal utilise le renderer logiciel (également appelé « WARP ») à la place du renderer matériel.

Nom de la propriété : `experimental.rendering.software`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Activer l'acrylique non focalisé

Lorsque ce paramètre est défini sur `true`, le terminal tentera d'utiliser de l'acrylique même lorsque la fenêtre n'est pas focalisée. Cela ne fonctionnera que si l'acrylique est activé dans le système d'exploitation et dans votre profil. Lorsqu'il est défini sur `false`, l'acrylique ne sera utilisé que lorsque la fenêtre est focalisée (même si `useAcrylic` est défini sur `true` dans le profil `unfocusedAppearance`).

Nom de la propriété : `compatibility.enableUnfocusedAcrylic`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`

> **ⓘ Important**
>
> Cette fonctionnalité n’est disponible que dans la **préversion de Terminal Windows**.

Actions personnalisées dans le Terminal Windows

Article • 23/05/2023

> **ⓘ Important**
>
> À partir du Terminal Windows version 1.4, le tableau `keybindings` est nommé `actions` dans le fichier settings.json. Le tableau `keybindings` est toujours pris en charge à des fins de compatibilité descendante, mais le terminal ne renomme pas automatiquement `keybindings` en `actions` dans votre **fichier settings.json**.

Vous pouvez créer des actions personnalisées dans le Terminal Windows qui vous permettent de contrôler la façon dont vous interagissez avec le Terminal. Ces actions sont automatiquement ajoutées à la palette de commandes.

Formats d'action

Les actions peuvent être structurées aux formats suivants :

Commandes sans arguments

JSON

```
{ "command": "commandName", "keys": "modifiers+key" }
```

Par exemple, ce paramètre par défaut utilise le raccourci `ALT + F4` pour fermer la fenêtre de terminal :

JSON

```
{ "command": "closeWindow", "keys": "alt+f4" }
```

Commandes avec arguments

JSON

```
{ "command": { "action": "commandName", "argument": "value" }, "keys":
"modifiers+key" }
```

Par exemple, ce paramètre par défaut utilise le raccourci `Ctrl + Maj + 1` pour ouvrir un nouvel onglet dans le terminal en fonction du profil qui figure en premier dans le menu déroulant (en général, le profil PowerShell s'ouvre) :

```JSON
{ "command": { "action": "newTab", "index": 0 }, "keys": "ctrl+shift+1" }
```

Propriétés des actions

Les actions peuvent être construites avec les propriétés suivantes.

Commande

Il s'agit de la commande exécutée lorsque les touches associées sont activées.

Nom de la propriété : `command`

Nécessité : Obligatoire

Accepte : String

Touches

Définit les combinaisons de touches utilisées pour appeler la commande. Les touches peuvent avoir un nombre quelconque de modificateurs avec une touche. Les modificateurs et les touches acceptés sont répertoriés ci-dessous.

Si l'action n'a pas de clés, elle s'affiche dans la palette de commandes, mais ne peut pas être appelée avec le clavier.

Nom de la propriété : `keys`

Nécessité : Facultatif

Accepte : Chaîne ou tableau[chaîne]

Action

Ajoute des fonctionnalités supplémentaires à certaines commandes.

Nom de la propriété : `action`

Nécessité : Facultatif

Accepte : String

Nom

Définit le nom qui s'affiche dans la palette de commandes. Si rien n'est indiqué, le terminal tente de générer automatiquement un nom.

Nom de la propriété : `name`

Nécessité : Facultatif

Accepte : String

Icône

Définit l'icône qui s'affiche dans la palette de commandes.

Nom de la propriété : `icon`

Nécessité : Facultatif

Accepte : Emplacement du fichier en tant que chaîne ou emoji

Modificateurs et touches acceptés

Modificateurs

`ctrl+`, `shift+`, `alt+`, `win+`

> ⓘ **Notes**
>
> Alors que la touche `Windows` est prise en charge en tant que touche de modification, le système réserve la plupart des combinaisons de touches `win+<key>`. Si le système d'exploitation a réservé cette combinaison de touches, le terminal ne recevra jamais cette combinaison.

Touches de modification

Type	Touches
Fonction et touches alphanumériques	`f1-f24`, `a-z`, `0-9`
symboles	`` ` ``, `plus`, `-`, `=`, `[`, `]`, `\`, `;`, `'`, `,`, `.`, `/`
Touches de direction	`down`, `left`, `right`, `up`, `pagedown`, `pageup`, `pgdn`, `pgup`, `end`, `home`
Touches d'action	`tab`, `enter`, `esc`, `escape`, `space`, `backspace`, `delete`, `insert`, `app`, `menu`
Touches du pavé numérique	`numpad_0-numpad_9`, `numpad0-numpad9`, `numpad_add`, `numpad_plus`, `numpad_decimal`, `numpad_period`, `numpad_divide`, `numpad_minus`, `numpad_subtract`, `numpad_multiply`
Touches du navigateur	`browser_back`, `browser_forward`, `browser_refresh`, `browser_stop`, `browser_search`, `browser_favorites`, `browser_home`

Remarque : `=` et `plus` sont équivalents. Attention à ne pas confondre cette dernière touche avec `numpad_plus`.

Commandes au niveau de l'application

Quitter

Ferme toutes les fenêtres du terminal ouvertes. Une boîte de dialogue de confirmation s'affiche dans la fenêtre active pour vérifier que vous souhaitez fermer toutes les fenêtres.

Nom de la commande : `quit`

Combinaison par défaut :

```JSON
{ "command": "quit" }
```

Fermer la fenêtre

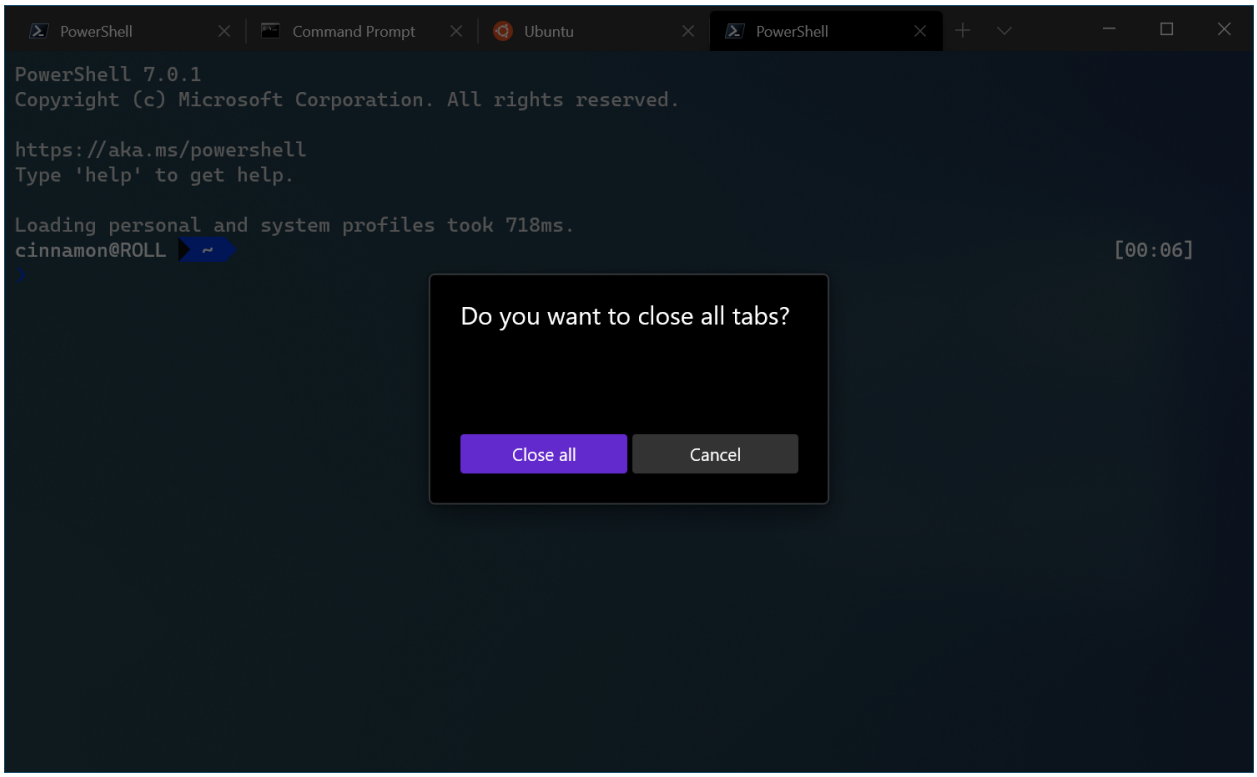
Ferme la fenêtre active et tous ses onglets. Si `confirmCloseAllTabs` est défini sur `true`, une boîte de dialogue de confirmation s'affiche pour vérifiez que vous souhaitez fermer tous les onglets. Pour plus d'informations sur ce paramètre, consultez la page Apparence.

Nom de la commande : `closeWindow`

Liaison par défaut :

JSON

```
{ "command": "closeWindow", "keys": "alt+f4" }
```



Rechercher

La boîte de dialogue Rechercher s'ouvre. Pour plus d'informations sur la recherche, consultez la page Rechercher.

Nom de la commande : `find`

Liaison par défaut :

JSON

```
{ "command": "find", "keys": "ctrl+shift+f" }
```

Rechercher le résultat de recherche suivant/précédent

Permet de parcourir les résultats de votre recherche.

Nom de la commande : `findMatch`

Copier les liaisons par défaut :

JSON

```
{ "command": { "action": "findMatch", "direction": "next" } },
{ "command": { "action": "findMatch", "direction": "prev" } }
```

Paramètres

Nom	Nécessité	Accepte	Description
`direction`	Obligatoire	`"next"`, `"prev"`	Sens de navigation dans les résultats de la recherche.

Ouvrir la liste déroulante

Le menu déroulant s'ouvre.

Nom de la commande : `openNewTabDropdown`

Liaison par défaut :

JSON

```
{ "command": "openNewTabDropdown", "keys": "ctrl+shift+space" }
```

Ouvrir les fichiers de paramètres

Ouvre l'interface utilisateur des paramètres, le fichier de paramètres personnalisés (settings.json) ou le fichier de paramètres par défaut (`defaults.json`) en fonction du champ `target`. Sans le champ `target`, le fichier de paramètres personnalisés s'ouvre.

Nom de la commande : `openSettings`

Copier les liaisons par défaut :

JSON

```
{ "command": { "action": "openSettings", "target": "settingsUI" }, "keys": "ctrl+," },
{ "command": { "action": "openSettings", "target": "settingsFile" }, "keys": "ctrl+shift+," },
{ "command": { "action": "openSettings", "target": "defaultsFile" }, "keys": "ctrl+alt+," },
```

Paramètres

Nom	Nécessité	Accepte	Description
`target`	Facultatif	`"settingsFile"`, `"defaultsFile"`, `"settingsUI"`, `"allFiles"`	Fichier de paramètres à ouvrir.

Ouvrir le menu système

Ouvre le menu système dans l'angle supérieur gauche de la fenêtre.

Nom de la commande : `openSystemMenu`

Liaison par défaut :

JSON

```
{ "command": "openSystemMenu", "keys": "alt+space" }
```

Passer en plein écran

Cela vous permet de basculer entre les tailles de fenêtre par défaut et plein écran.

Nom de la commande : `toggleFullscreen`

Copier les liaisons par défaut :

JSON

```
{ "command": "toggleFullscreen", "keys": "alt+enter" },
{ "command": "toggleFullscreen", "keys": "f11" }
```

Activer/désactiver le mode Focus

Vous permet d'entrer en « mode Focus », qui masque les onglets et la barre de titre.

Nom de la commande : `toggleFocusMode`

Liaison par défaut :

```JSON
{ "command": "toggleFocusMode" }
```

Activer/désactiver le mode Toujours visible

Vous permet d'activer ou de désactiver l'état « Toujours visible » de la fenêtre. En mode « Toujours visible », la fenêtre s'affiche devant toutes les autres fenêtres qui ne sont pas au premier plan.

Nom de la commande : `toggleAlwaysOnTop`

Liaison par défaut :

```JSON
{ "command": "toggleAlwaysOnTop" }
```

Envoyer l'entrée

Envoie une entrée de texte arbitraire au shell. À titre d'exemple, l'entrée `"text\n"` écrit « text » suivi d'un saut de ligne dans le shell.

Les séquences d'échappement ANSI peuvent être utilisées, mais les codes d'échappement comme `\x1b` doivent être écrits sous la forme `\u001b`. Par exemple, `"\u001b[A"` se comporte comme si le bouton flèche haut avait été enfoncé.

Nom de la commande : `sendInput`

Liaison par défaut :

Cette commande n'est pas actuellement liée dans les paramètres par défaut.

```JSON
{ "command": { "action": "sendInput", "input": "\u001b[A" }, "keys": "" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`input`	Obligatoire	String	Entrée de texte à alimenter dans le shell.

Commandes de gestion des onglets

Fermer l'onglet

Ferme l'onglet à un index donné. Si aucun index n'est fourni, utilise l'index de l'onglet ayant le focus.

Nom de la commande : `closeTab`

Paramètres

Nom	Nécessité	Accepte	Description
`index`	Facultatif	Integer	Position de l'onglet à fermer.

Fermer tous les autres onglets

Ferme tous les onglets sauf celui à un index donné. Si aucun index n'est fourni, utilise l'index de l'onglet ayant le focus.

Nom de la commande : `closeOtherTabs`

Liaison par défaut :

JSON

```
{ "command": "closeOtherTabs" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`index`	Facultatif	Integer	Position de l'onglet à garder ouvert.

Fermer les onglets après l'index

Ferme les onglets situés après l'onglet à un index donné. Si aucun index n'est fourni, utilise l'index de l'onglet ayant le focus.

Nom de la commande : `closeTabsAfter`

Liaison par défaut :

```JSON
{ "command": "closeTabsAfter" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`index`	Facultatif	Integer	Position du dernier onglet à garder ouvert.

Dupliquer l'onglet

Fait une copie du profil et du répertoire de l'onglet actuel et l'ouvre. Les VARIABLES ENV modifiées/ajoutées ne sont pas incluses.

Nom de la commande : `duplicateTab`

Liaison par défaut :

```JSON
{ "command": "duplicateTab", "keys": "ctrl+shift+d" }
```

Nouvel onglet

Cela crée un nouvel onglet. Sans aucun argument, cela ouvrira le profil par défaut dans un nouvel onglet. Si aucun index n'est spécifié, le paramètre équivalent du profil par défaut sera utilisé. Si l'index ne correspond pas à un profil, les clés sont transmises directement au terminal (ou ignorées si aucune clé n'a été utilisée pour invoquer l'action).

Nom de la commande : `newTab`

Copier les liaisons par défaut :

```JSON
```

```
{ "command": "newTab", "keys": "ctrl+shift+t" },
{ "command": { "action": "newTab", "index": 0 }, "keys": "ctrl+shift+1" },
{ "command": { "action": "newTab", "index": 1 }, "keys": "ctrl+shift+2" },
{ "command": { "action": "newTab", "index": 2 }, "keys": "ctrl+shift+3" },
{ "command": { "action": "newTab", "index": 3 }, "keys": "ctrl+shift+4" },
{ "command": { "action": "newTab", "index": 4 }, "keys": "ctrl+shift+5" },
{ "command": { "action": "newTab", "index": 5 }, "keys": "ctrl+shift+6" },
{ "command": { "action": "newTab", "index": 6 }, "keys": "ctrl+shift+7" },
{ "command": { "action": "newTab", "index": 7 }, "keys": "ctrl+shift+8" },
{ "command": { "action": "newTab", "index": 8 }, "keys": "ctrl+shift+9" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`commandline`	Facultatif	Nom de fichier exécutable sous forme de chaîne	Exécutable exécuté dans l'onglet.
`startingDirectory`	Facultatif	Emplacement du dossier sous forme de chaîne	Répertoire dans lequel l'onglet s'ouvre.
`elevate`	Facultatif	`true`, `false`, `null`	Remplace la propriété elevate du profil. En cas d'omission, cette action se comporte conformément au paramètre `elevate` du profil. Quand la valeur est `true` ou `false`, cette action se comporte comme si le profil était défini avec `"elevate": true` ou `"elevate": false` (respectivement).
`tabTitle`	Facultatif	String	Le titre du nouvel onglet.
`index`	Facultatif	Entier	Le profil qui s'ouvre en fonction de sa position dans la liste déroulante (à partir de 0).
`profile`	Facultatif	Nom ou GUID du profil sous forme de chaîne	Le profil qui s'ouvre en fonction de son GUID ou de son nom.

Nom	Nécessité	Accepte	Description
`colorScheme`	Facultatif	Nom du modèle de couleurs en tant que chaîne	Modèle à utiliser à la place du `colorScheme` défini pour le profil
`suppressApplicationTitle`	Facultatif	`true`, `false`	Quand la valeur est `false`, les applications peuvent modifier le titre de l'onglet en envoyant des messages de modification de titre. Quand la valeur est `true`, ces messages sont supprimés. Si rien n'est indiqué, le comportement est hérité des paramètres du profil. Pour entrer un nouveau titre d'onglet et le rendre persistant, cette valeur doit être true.

Ouvrir l'onglet suivant

Ouvre l'onglet à droite de l'onglet actuel.

Nom de la commande : `nextTab`

Liaison par défaut :

```JSON
{ "command": "nextTab", "keys": "ctrl+tab" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`tabSwitcherMode`	Facultatif	`"mru"`, `"inOrder"`, `"disabled"`	Permet de passer à l'onglet suivant avec `"tabSwitcherMode"`. Si aucun mode n'est fourni, utilisez celui défini globalement.

Ouvrir l'onglet précédent

Ouvre l'onglet à gauche de l'onglet actuel.

Nom de la commande : `prevTab`

Liaison par défaut :

JSON

```
{ "command": "prevTab", "keys": "ctrl+shift+tab" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`tabSwitcherMode`	Facultatif	`"mru"`, `"inOrder"`, `"disabled"`	Permet de passer à l'onglet précédent avec `"tabSwitcherMode"`. Si aucun mode n'est fourni, utilisez celui défini globalement.

Recherche d'onglet

Ouvre la zone de recherche d'onglet.

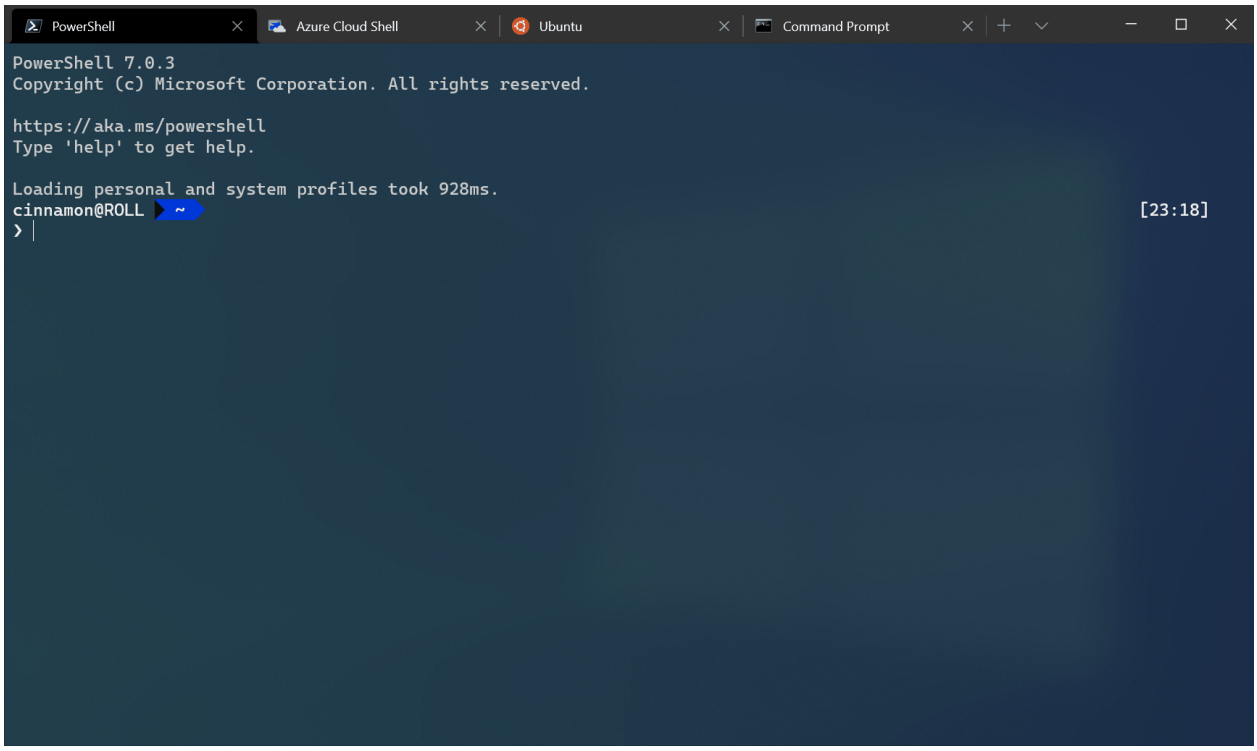
Nom de la commande : `tabSearch`

Liaison par défaut :

Cette commande n'est pas actuellement liée dans les paramètres par défaut.

JSON

```
{"command": "tabSearch", "keys": ""}
```



Ouvrir un onglet spécifique

Ouvre un onglet spécifique en fonction de l'index.

Nom de la commande : `switchToTab`

Copier les liaisons par défaut :

JSON

```
{ "command": { "action": "switchToTab", "index": 0 }, "keys": "ctrl+alt+1" },
{ "command": { "action": "switchToTab", "index": 1 }, "keys": "ctrl+alt+2" },
{ "command": { "action": "switchToTab", "index": 2 }, "keys": "ctrl+alt+3" },
{ "command": { "action": "switchToTab", "index": 3 }, "keys": "ctrl+alt+4" },
{ "command": { "action": "switchToTab", "index": 4 }, "keys": "ctrl+alt+5" },
{ "command": { "action": "switchToTab", "index": 5 }, "keys": "ctrl+alt+6" },
{ "command": { "action": "switchToTab", "index": 6 }, "keys": "ctrl+alt+7" },
{ "command": { "action": "switchToTab", "index": 7 }, "keys": "ctrl+alt+8" },
{ "command": { "action": "switchToTab", "index": 8 }, "keys": "ctrl+alt+9" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`index`	Obligatoire	Entier	L'onglet qui s'ouvre en fonction de sa position dans la barre d'onglets (à partir de 0).

Renommer un onglet

Cette commande peut être utilisée pour renommer un onglet en une chaîne spécifique.

Nom de la commande : `renameTab`

Liaison par défaut :

Cette commande n'est pas actuellement liée dans les paramètres par défaut.

JSON

```
// Rename a tab to "Foo"
{ "command": { "action": "renameTab", "title": "Foo" }, "keys": "" }

// Reset the tab's name
{ "command": { "action": "renameTab", "title": null }, "keys": "" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`title`	Facultatif	String	Nouveau titre à utiliser pour cet onglet. En cas d'omission, cette commande rétablit le titre d'origine de l'onglet.

Ouvrir la zone de texte de renommage d'un onglet

Cette commande remplace le titre de l'onglet par un champ de texte qui vous permet de modifier le titre de l'onglet actif. Si vous effacez le champ de texte, le titre d'onglet par défaut est rétabli pour l'instance de shell active.

Nom de la commande : `openTabRenamer`

Liaison par défaut :

JSON

```
{ "command": "openTabRenamer" }
```

Modifier la couleur de l'onglet

Cette commande peut être utilisée pour remplacer la couleur d'un onglet par une valeur spécifique.

Nom de la commande : `setTabColor`

Liaison par défaut :

Cette commande n'est pas actuellement liée dans les paramètres par défaut.

JSON

```
// Change the tab's color to a bright magenta
{ "command": { "action": "setTabColor", "color": "#ff00ff" }, "keys": "" }

// Reset the tab's color
{ "command": { "action": "setTabColor", "color": null }, "keys": "" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`color`	Facultatif	Chaîne au format hexadécimal : `"#rgb"` ou `"#rrggbb"`	Nouvelle couleur à utiliser pour cet onglet. En cas d'omission, cette commande rétablit la couleur d'origine de l'onglet.

Ouvrir le sélecteur de couleurs d'onglet

Vous pouvez utiliser cette commande pour ouvrir le sélecteur de couleurs pour l'onglet actif. Le sélecteur permet de définir une couleur pour l'onglet au moment de l'exécution.

Nom de la commande : `openTabColorPicker`

Liaison par défaut :

JSON

```
{ "command": "openTabColorPicker" }
```

Déplacer l'onglet

Cette commande déplace l'onglet vers l'arrière (« backward ») et vers l'avant (« forward »), ce qui équivaut à « left » et « right » dans l'interface utilisateur de gauche à droite.

Nom de la commande : `moveTab`

Liaison par défaut :

```JSON
// Move tab backward (left in LTR)
{ "command": { "action": "moveTab", "direction": "backward" } }

// Move tab forward (right in LTR)
{ "command": { "action": "moveTab", "direction": "forward" } }
```

Paramètres

Nom	Nécessité	Accepte	Description
`direction`	Obligatoire	`"backward"`, `"forward"`	Direction dans laquelle l'onglet est déplacé.

Commandes de gestion des fenêtres

Nouvelle fenêtre

Crée une fenêtre. Si aucun argument n'est spécifié, le profil par défaut s'ouvre dans une nouvelle fenêtre (quel que soit la valeur du paramètre `windowingBehavior`). Si aucune action n'est spécifiée, le paramètre équivalent du profil par défaut est utilisé.

Nom de la commande : `newWindow`

Copier les liaisons par défaut :

```JSON
{ "command": "newWindow", "keys": "ctrl+shift+n" },
```

Paramètres

Nom	Nécessité	Accepte	Description
`commandline`	Facultatif	Nom de fichier exécutable sous forme de chaîne	Exécutable exécuté dans l'onglet.
`startingDirectory`	Facultatif	Emplacement du dossier sous forme de chaîne	Répertoire dans lequel la fenêtre s'ouvre.
`tabTitle`	Facultatif	String	Titre de l'onglet de la fenêtre.
`index`	Facultatif	Entier	Le profil qui s'ouvre en fonction de sa position dans la liste déroulante (à partir de 0).
`profile`	Facultatif	Nom ou GUID du profil sous forme de chaîne	Le profil qui s'ouvre en fonction de son GUID ou de son nom.
`suppressApplicationTitle`	Facultatif	`true`, `false`	Quand la valeur est `false`, les applications peuvent modifier le titre de l'onglet en envoyant des messages de modification de titre. Quand la valeur est `true`, ces messages sont supprimés. Si rien n'est indiqué, le comportement est hérité des paramètres du profil.

Renommer une fenêtre

Cette commande peut être utilisée pour renommer une fenêtre en une chaîne spécifique.

Nom de la commande : `renameWindow`

Liaison par défaut :

Cette commande n'est pas actuellement liée dans les paramètres par défaut.

JSON

```
// Rename a window to "Foo"
{ "command": { "action": "renameWindow", "name": "Foo" }, "keys": "" }
```

```
// Reset the window's name
{ "command": { "action": "renameWindow", "name": null }, "keys": "" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`name`	Facultatif	String	Nouveau nom à utiliser pour cette fenêtre. En cas d'omission, cette commande rétablit le nom d'origine de la fenêtre.

Ouvrir la boîte de dialogue de renommage de fenêtre

Cette commande affiche une fenêtre contextuelle qui vous permet de modifier le nom de la fenêtre active. L'effacement du champ de texte réinitialise le nom de la fenêtre.

Nom de la commande : `openWindowRenamer`

Liaison par défaut :

JSON

```
{ "command": "openWindowRenamer" }
```

Identifier la fenêtre

Ouvre une fenêtre de superposition sur la fenêtre qui a le focus pour afficher le nom et l'index de la fenêtre.

Nom de la commande : `identifyWindow`

Liaison par défaut :

JSON

```
{"command": "identifyWindow", "keys": "" },
```

Identifier les fenêtres

Ouvre une fenêtre de superposition sur toutes les fenêtre pour afficher le nom et l'index de chaque fenêtre.

Nom de la commande : `identifyWindows`

Liaison par défaut :

Cette commande n'est pas actuellement liée dans les paramètres par défaut.

JSON

```
{"command": "identifyWindows" },
```

Commandes de gestion du volet

Fractionner un volet

Divise par deux la taille du volet actif et en ouvre un autre. Sans aucun argument, le profil par défaut s'ouvre dans le nouveau volet. Si aucune action n'est spécifiée, le paramètre équivalent du profil par défaut est utilisé.

Nom de la commande : `splitPane`

Copier les liaisons par défaut :

JSON

```
// In settings.json
{ "command": { "action": "splitPane", "split": "auto", "splitMode":
"duplicate" }, "keys": "alt+shift+d" },

// In defaults.json
{ "command": { "action": "splitPane", "split": "horizontal" }, "keys":
"alt+shift+-" },
{ "command": { "action": "splitPane", "split": "vertical" }, "keys":
"alt+shift+plus" },
{ "command": { "action": "splitPane", "split": "up" } },
{ "command": { "action": "splitPane", "split": "right" } },
{ "command": { "action": "splitPane", "split": "down" } },
{ "command": { "action": "splitPane", "split": "left" } }
```

Paramètres

Nom	Nécessité	Accepte	Description

Nom	Nécessité	Accepte	Description
`split`	Obligatoire	`"vertical"`, `"horizontal"`, `"auto"`, `"up"`, `"right"`, `"down"`, `"left"`	Mode de fractionnement du volet. `"auto"` fractionne dans le sens qui fournit la plus grande surface.
`commandline`	Facultatif	Nom de fichier exécutable sous forme de chaîne	Exécutable exécuté dans le volet.
`startingDirectory`	Facultatif	Emplacement du dossier sous forme de chaîne	Répertoire dans lequel le volet s'ouvre.
`elevate`	Facultatif	`true`, `false`, `null`	Remplace la propriété elevate du profil. En cas d'omission, cette action se comporte conformément au paramètre `elevate` du profil. Quand la valeur est `true` ou `false`, cette action se comporte comme si le profil était défini avec `"elevate": true` ou `"elevate": false` (respectivement).
`tabTitle`	Facultatif	String	Titre de l'onglet lorsque le nouveau volet a le focus.
`index`	Facultatif	Entier	Le profil qui s'ouvre en fonction de sa position dans la liste déroulante (à partir de 0).
`profile`	Facultatif	Nom ou GUID du profil sous forme de chaîne	Le profil qui s'ouvre en fonction de son GUID ou de son nom.
`colorScheme`	Facultatif	Nom du modèle de couleurs en tant que chaîne	Modèle à utiliser à la place du `colorScheme` défini pour le profil

Nom	Nécessité	Accepte	Description
`suppressApplicationTitle`	Facultatif	`true`, `false`	Quand la valeur est `false`, les applications peuvent modifier le titre de l'onglet en envoyant des messages de modification de titre. Quand la valeur est `true`, ces messages sont supprimés. Si rien n'est indiqué, le comportement est hérité des paramètres du profil.
`splitMode`	Facultatif	`"duplicate"`	Contrôle le mode de fractionnement du volet. Accepte uniquement `"duplicate"`, qui duplique le profil du volet actif dans un nouveau volet.
`size`	Facultatif	Float	Spécifiez la taille du nouveau volet en tant que fraction de la taille du volet actif. `1.0` signifie « tout du volet actuel » et `0.0` signifie « rien du parent ». La valeur par défaut est `0.5`.

Fermer le volet

Ferme le volet actif. S'il n'existe pas de volets fractionnés, l'onglet actuel est fermé. Si un seul onglet est ouvert, la fenêtre est fermée.

Nom de la commande : `closePane`

Liaison par défaut :

JSON

```
{ "command": "closePane", "keys": "ctrl+shift+w" }
```

Déplacer le focus du volet

Modifie le focus sur un autre volet en fonction de la direction. La définition de `direction` sur `"previous"` déplace le focus vers le dernier volet utilisé.

Nom de la commande : `moveFocus`

Copier les liaisons par défaut :

JSON

```
{ "command": { "action": "moveFocus", "direction": "down" }, "keys":
"alt+down" },
{ "command": { "action": "moveFocus", "direction": "left" }, "keys":
"alt+left" },
{ "command": { "action": "moveFocus", "direction": "right" }, "keys":
"alt+right" },
{ "command": { "action": "moveFocus", "direction": "up" }, "keys": "alt+up"
},
{ "command": { "action": "moveFocus", "direction": "previous" }, "keys":
"ctrl+alt+left" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`direction`	Obligatoire	`"left"`, `"right"`, `"up"`, `"down"`, `"previous"`, `"previousInOrder"`, `"nextInOrder"`, `"first"`, `"parent"`, `"child"`	Direction dans laquelle le focus se déplace.

Valeurs `direction` acceptées

- `up`, `down`, `left` ou `right` : déplace le focus dans la direction donnée.
- `first` : déplace le focus sur le premier volet de nœud terminal dans l'arborescence.
- `previous` : déplace le focus sur le dernier volet utilisé avant le volet actif.
- `nextInOrder` ou `previousInOrder` : déplace le focus sur le volet suivant ou précédent selon l'ordre de création.
- `parent` : déplace le focus pour sélectionner le volet parent du volet actif. Cela permet à l'utilisateur de sélectionner plusieurs volets à la fois.
- `child` : déplace le focus sur le premier volet enfant de ce volet.

Volet Déplacer

Permet de déplacer le volet actif vers un autre onglet de la fenêtre.

Nom de la commande : `movePane`

Liaisons par défaut : *(aucune)*

Paramètres

Nom	Nécessité	Accepte	Description

Nom	Nécessité	Accepte	Description
`index`	Obligatoire	nombre	Index indexé à zéro de l'onglet vers lequel se déplacer.

Permuter les volets

Permet de permuter la position de deux volets dans un onglet. L'opération porte sur le volet actif et un volet cible désigné par le paramètre `direction`.

Nom de la commande : `moveFocus`

Copier les liaisons par défaut :

JSON

```
{ "command": { "action": "swapPane", "direction": "down" } },
{ "command": { "action": "swapPane", "direction": "left" } },
{ "command": { "action": "swapPane", "direction": "right" } },
{ "command": { "action": "swapPane", "direction": "up" } },
{ "command": { "action": "swapPane", "direction": "previous"} },
{ "command": { "action": "swapPane", "direction": "previousInOrder"} },
{ "command": { "action": "swapPane", "direction": "nextInOrder"} },
{ "command": { "action": "swapPane", "direction": "first" } },
```

Paramètres

Nom	Nécessité	Accepte	Description
`direction`	Obligatoire	`"left"`, `"right"`, `"up"`, `"down"`, `"previous"`, `"previousInOrder"`, `"nextInOrder"`, `"first"`, `"parent"`, `"child"`	Direction dans laquelle le focus se déplace.

Valeurs `direction` acceptées (valeurs identiques à celles de la commande `moveFocus`)

- `up`, `down`, `left` ou `right` : permute le volet actif avec le volet situé dans la direction donnée.
- `first` : permute le volet actif avec le premier volet de nœud terminal de l'arborescence.
- `previous` : permute le volet actif avec le dernier volet utilisé avant le volet actif.
- `nextInOrder` ou `previousInOrder` : permute le volet actif avec le volet suivant ou précédent selon l'ordre de création.
- `parent` : ne fait rien.
- `child` : ne fait rien.

Zoomer sur un volet

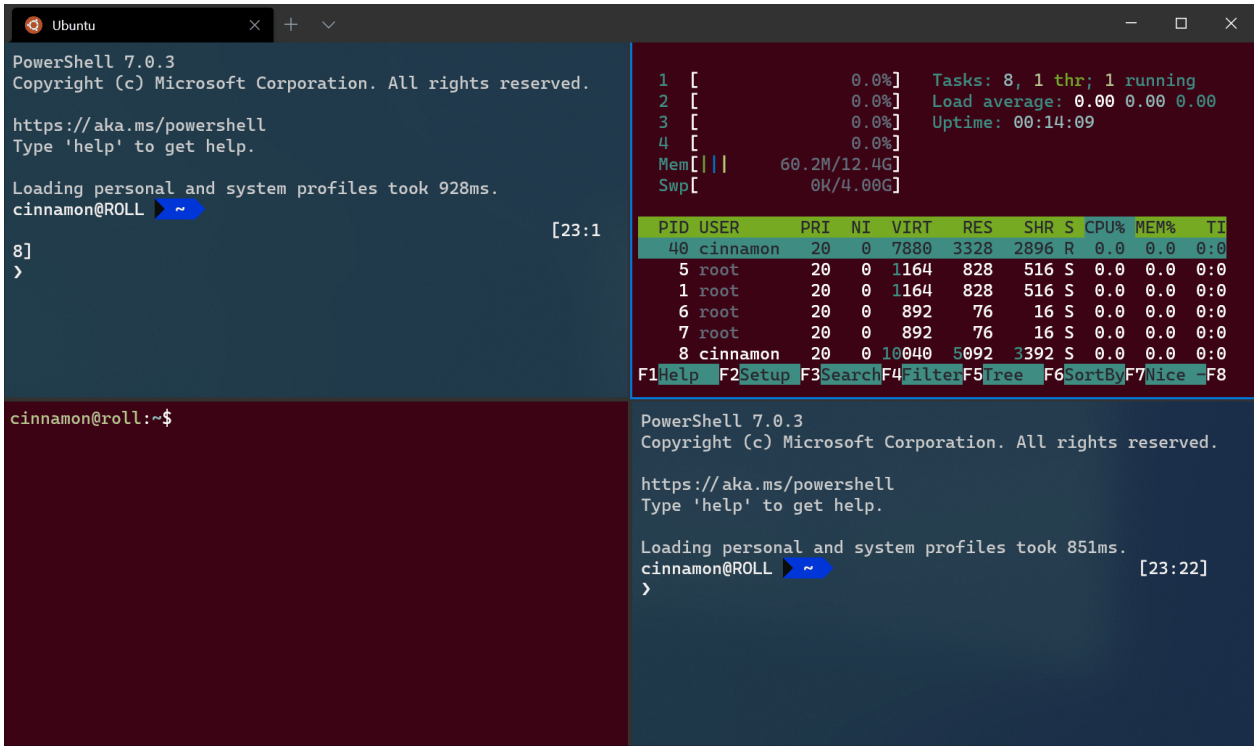
Développe le volet ayant le focus pour remplir tout le contenu de la fenêtre.

Nom de la commande : `togglePaneZoom`

Liaison par défaut :

JSON

```
{ "command": "togglePaneZoom" }
```



Redimensionner un volet

Modifie la taille du volet actif.

Nom de la commande : `resizePane`

Copier les liaisons par défaut :

JSON

```
{ "command": { "action": "resizePane", "direction": "down" }, "keys":
"alt+shift+down" },
{ "command": { "action": "resizePane", "direction": "left" }, "keys":
"alt+shift+left" },
{ "command": { "action": "resizePane", "direction": "right" }, "keys":
"alt+shift+right" },
```

```
{ "command": { "action": "resizePane", "direction": "up" }, "keys":
"alt+shift+up" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`direction`	Obligatoire	`"left"`, `"right"`, `"up"`, `"down"`	Sens dans lequel le volet sera redimensionné.

Marquage d'un volet en lecture seule

Vous pouvez marquer un volet comme étant en lecture seule, ce qui a pour effet d'empêcher le transfert de toute entrée dans la mémoire tampon de texte. Si vous tentez de fermer ou d'entrer du texte dans un volet en lecture seule, le terminal affiche un avertissement contextuel.

Nom de la commande : `toggleReadOnlyMode`

Copier les liaisons par défaut :

JSON

```
{ "command": "toggleReadOnlyMode" }
```

Vous pouvez activer le mode lecture seule dans un volet. Cela fonctionne de la même façon qu'un bouton bascule, mais par contre, cela ne change pas l'état si redéclenché.

Nom de la commande : `enableReadOnlyMode`

Copier les liaisons par défaut :

JSON

```
{ "command": "enableReadOnlyMode" }
```

> **ⓘ Important**
>
> Cette fonctionnalité n'est disponible que dans la **préversion de Terminal Windows**.

Vous pouvez désactiver le mode lecture seule dans un volet. Cela fonctionne de la même façon qu'un bouton bascule, mais par contre, cela ne change pas l'état si redéclenché.

Nom de la commande : `disableReadOnlyMode`

Copier les liaisons par défaut :

JSON

```
{ "command": "disableReadOnlyMode" }
```

> **ⓘ Important**
>
> Cette fonctionnalité n'est disponible que dans la **préversion de Terminal Windows**.

Fractionner un volet

Divise par deux la taille du volet actif et en ouvre un autre. Sans aucun argument, le profil par défaut s'ouvre dans le nouveau volet. Si aucune action n'est spécifiée, le paramètre équivalent du profil par défaut est utilisé.

Nom de la commande : `splitPane`

Copier les liaisons par défaut :

JSON

```
// In settings.json
{ "command": { "action": "splitPane", "split": "auto", "splitMode":
"duplicate" }, "keys": "alt+shift+d" },

// In defaults.json
{ "command": { "action": "splitPane", "split": "horizontal" }, "keys":
"alt+shift+-" },
{ "command": { "action": "splitPane", "split": "vertical" }, "keys":
"alt+shift+plus" },
{ "command": { "action": "splitPane", "split": "up" } },
{ "command": { "action": "splitPane", "split": "right" } },
{ "command": { "action": "splitPane", "split": "down" } },
{ "command": { "action": "splitPane", "split": "left" } }
```

Paramètres

Nom	Nécessité	Accepte	Description
`split`	Obligatoire	`"vertical"`, `"horizontal"`, `"auto"`, `"up"`, `"right"`, `"down"`, `"left"`	Mode de fractionnement du volet. `"auto"` fractionne dans le sens qui fournit la plus grande surface.
`commandline`	Facultatif	Nom de fichier exécutable sous forme de chaîne	Exécutable exécuté dans le volet.
`startingDirectory`	Facultatif	Emplacement du dossier sous forme de chaîne	Répertoire dans lequel le volet s'ouvre.
`elevate`	Facultatif	`true`, `false`, `null`	Remplace la propriété elevate du profil. En cas d'omission, cette action se comporte conformément au paramètre `elevate` du profil. Quand la valeur est `true` ou `false`, cette action se comporte comme si le profil était défini avec `"elevate": true` ou `"elevate": false` (respectivement).
`tabTitle`	Facultatif	String	Titre de l'onglet lorsque le nouveau volet a le focus.
`index`	Facultatif	Entier	Le profil qui s'ouvre en fonction de sa position dans la liste déroulante (à partir de 0).
`profile`	Facultatif	Nom ou GUID du profil sous forme de chaîne	Le profil qui s'ouvre en fonction de son GUID ou de son nom.
`colorScheme`	Facultatif	Nom du modèle de couleurs en tant que chaîne	Modèle à utiliser à la place du `colorScheme` défini pour le profil

Nom	Nécessité	Accepte	Description
`suppressApplicationTitle`	Facultatif	`true`, `false`	Quand la valeur est `false`, les applications peuvent modifier le titre de l'onglet en envoyant des messages de modification de titre. Quand la valeur est `true`, ces messages sont supprimés. Si rien n'est indiqué, le comportement est hérité des paramètres du profil.
`splitMode`	Facultatif	`"duplicate"`	Contrôle le mode de fractionnement du volet. Accepte uniquement `"duplicate"`, qui duplique le profil du volet actif dans un nouveau volet.
`size`	Facultatif	Float	Spécifiez la taille du nouveau volet en tant que fraction de la taille du volet actif. `1.0` signifie « tout du volet actuel » et `0.0` signifie « rien du parent ». La valeur par défaut est `0.5`.

Commandes d'intégration du presse-papiers

Copier

Cette option permet de copier le contenu du terminal sélectionné dans le presse-papiers. Si aucune sélection n'existe, la pression simultanée sur les touches est envoyée directement au terminal.

Nom de la commande : `copy`

Copier les liaisons par défaut :

JSON

```
// In settings.json
{ "command": { "action": "copy", "singleLine": false }, "keys": "ctrl+c" },

// In defaults.json
{ "command": { "action": "copy", "singleLine": false }, "keys":
"ctrl+shift+c" },
{ "command": { "action": "copy", "singleLine": false }, "keys":
"ctrl+insert" },
{ "command": { "action": "copy", "singleLine": false }, "keys": "enter" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`singleLine`	Facultatif	`true`, `false`	Si `true`, le contenu copié est copié sous la forme d'une ligne unique. Si `false`, les nouvelles lignes sont conservées à partir du texte sélectionné.
`copyFormatting`	Facultatif	`true`, `false`, `"all"`, `"none"`, `"html"`, `"rtf"`	Quand la valeur est `true`, la mise en forme de la couleur et de la police du texte sélectionné est également copiée dans votre Presse-papiers. Quand la valeur est `false`, seul le texte brut est copié dans votre Presse-papiers. Vous pouvez également spécifier les formats que vous souhaitez copier. Quand la valeur est `null`, le comportement global de `"copyFormatting"` est hérité.

Coller

Insère le contenu qui a été copié dans le presse-papiers.

Nom de la commande : `paste`

Copier les liaisons par défaut :

JSON

```
// In settings.json
{ "command": "paste", "keys": "ctrl+v" },

// In defaults.json
{ "command": "paste", "keys": "ctrl+shift+v" },
{ "command": "paste", "keys": "shift+insert" }
```

Étendre la sélection au mot

Si une sélection existe, cela développe la sélection pour englober entièrement les mots partiellement sélectionnés.

Nom de la commande : `expandSelectionToWord`

Copier les liaisons par défaut :

JSON

```
{ "command": "expandSelectionToWord" }
```

Sélectionner tout

Cela sélectionne tout le contenu du tampon de texte.

Nom de la commande : `selectAll`

Copier les liaisons par défaut :

```JSON
{ "command": "selectAll", "keys": "ctrl+shift+a" }
```

Mode Marque

Cela active ou désactive le mode marquage. Le mode marquage est un mode qui vous permet d'utiliser le clavier pour faire une sélection à la position du curseur dans le terminal.

Nom de la commande : `markMode`

Copier les liaisons par défaut :

```JSON
{ "command": "markMode", "keys": "ctrl+shift+m" },
```

Passer à un autre marqueur de sélection

Lorsque vous modifiez une sélection à l'aide du clavier, vous déplacez une extrémité de la sélection. Vous pouvez utiliser cette action pour passer à l'autre marqueur de sélection.

Nom de la commande : `switchSelectionEndpoint`

Copier les liaisons par défaut :

```JSON
{ "command": "switchSelectionEndpoint" },
```

Activer/Désactiver la sélection de bloc

Fait de la sélection existante une sélection de bloc, où la zone sélectionnée est un rectangle qui n'inclut pas le début et la fin de chaque ligne.

Nom de la commande : `toggleBlockSelection`

Copier les liaisons par défaut :

```JSON
{ "command": "toggleBlockSelection" },
```

Commandes de défilement

Faire défiler vers le haut

Fait défiler l'écran vers le haut du nombre de lignes défini par `"rowsToScroll"`. Si `"rowsToScroll"` n'est pas fourni, le défilement vers le haut est défini par la valeur système par défaut, qui correspond au défilement avec la souris.

Nom de la commande : `scrollUp`

Liaison par défaut :

```JSON
{ "command": "scrollUp", "keys": "ctrl+shift+up" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`rowsToScroll`	Facultatif	Integer	Nombre de lignes à faire défiler.

Faire défiler vers le bas

Fait défiler l'écran vers le bas du nombre de lignes défini par `"rowsToScroll"`. Si `"rowsToScroll"` n'est pas fourni, le défilement vers le bas est défini par la valeur système par défaut, qui correspond au défilement avec la souris.

Nom de la commande : `scrollDown`

Liaison par défaut :

```JSON
{ "command": "scrollDown", "keys": "ctrl+shift+down" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`rowsToScroll`	Facultatif	Integer	Nombre de lignes à faire défiler.

Faire défiler une page entière

Fait défiler l'écran d'une page entière, ce qui correspond à la hauteur de la fenêtre.

Nom de la commande : `scrollUpPage`

Liaison par défaut :

```JSON
{ "command": "scrollUpPage", "keys": "ctrl+shift+pgup" }
```

Faire défiler l'intégralité d'une page

Fait défiler l'écran d'une page entière, qui est la hauteur de la fenêtre.

Nom de la commande : `scrollDownPage`

Liaison par défaut :

```JSON
{ "command": "scrollDownPage", "keys": "ctrl+shift+pgdn" }
```

Faire défiler jusqu'à l'historique le plus ancien

Fait défiler l'écran jusqu'en haut de la mémoire tampon d'entrée.

Nom de la commande : `scrollToTop`

Liaison par défaut :

```JSON
{ "command": "scrollToTop", "keys": "ctrl+shift+home" }
```

Faire défiler jusqu'au dernier historique

Fait défiler l'écran jusqu'en bas de la mémoire tampon d'entrée.

Nom de la commande : `scrollToBottom`

Liaison par défaut :

```JSON
{ "command": "scrollToBottom", "keys": "ctrl+shift+end" }
```

Vider la mémoire tampon

Cette action peut être utilisée pour effacer manuellement la mémoire tampon du terminal. Elle est utile lorsque vous ne disposez pas d'une invite d'interpréteur de ligne de commande et que vous ne pouvez pas facilement exécuter `Clear-Host`/`cls`/`clear`.

Nom de la commande : `clearBuffer`

Copier les liaisons par défaut :

```JSON
{ "command": { "action": "clearBuffer", "clear": "all" } }
```

Paramètres

Nom	Nécessité	Accepte	Description

Nom	Nécessité	Accepte	Description
`clear`	Facultatif	`"screen"`, `"scrollback"`, `"all"`	Partie de l'écran à effacer. • `"screen"` : effacer le contenu de la fenêtre d'affichage du terminal. Laisse le défilement arrière intact. Déplace la ligne de curseur en haut de la fenêtre d'affichage (inchangée). • `"scrollback"` : effacer le défilement arrière. Laisse la fenêtre d'affichage intacte. • `"all"` (*par défaut*) : effacer le défilement arrière et la fenêtre d'affichage visible. Déplace la ligne de curseur en haut de la fenêtre d'affichage.

Commandes d'ajustement visuel

Ajuster la taille de police

Modifie la taille du texte d'une valeur de point spécifiée.

Nom de la commande : `adjustFontSize`

Copier les liaisons par défaut :

JSON

```
{ "command": { "action": "adjustFontSize", "delta": 1 }, "keys": "ctrl+=" },
{ "command": { "action": "adjustFontSize", "delta": -1 }, "keys": "ctrl+-"
},
{ "command": { "action": "adjustFontSize", "delta": 1 }, "keys":
"ctrl+numpad_plus" },
{ "command": { "action": "adjustFontSize", "delta": -1 }, "keys":
"ctrl+numpad_minus" }
```

Paramètres

Nom	Nécessité	Accepte	Description
`delta`	Obligatoire	Entier	Quantité de changement de taille par appel de commande.

Réinitialiser la taille de police

Rétablit la valeur par défaut de la taille du texte.

Nom de la commande : `resetFontSize`

Copier les liaisons par défaut :

JSON

```
{ "command": "resetFontSize", "keys": "ctrl+0" },
{ "command": "resetFontSize", "keys": "ctrl+numpad_0" }
```

Ajuster l'opacité

Permet de modifier l'opacité de la fenêtre. Si `relative` a la valeur true, l'opacité est ajustée par rapport à l'opacité actuelle. Sinon, l'opacité est définie directement avec la valeur `opacity` donnée.

Nom de la commande : `adjustOpacity`

Copier les liaisons par défaut :

JSON

```
{ "command": { "action": "adjustOpacity", "relative": false, "opacity": 0 }
},
{ "command": { "action": "adjustOpacity", "relative": false, "opacity": 25 }
},
{ "command": { "action": "adjustOpacity", "relative": false, "opacity": 50 }
},
{ "command": { "action": "adjustOpacity", "relative": false, "opacity": 100
} }
```

Paramètres

Nom	Nécessité	Accepte	Description
`opacity`	Facultatif	Integer	Degré d'opacité à affecter au terminal ou montant du changement de l'opacité, en fonction de la valeur de `relative`
`relative`	Facultatif	Booléen	Si la valeur est true, l'opacité actuelle est ajustée selon le paramètre `opacity` donné. Si la valeur est false, l'opacité est définie avec cette valeur.

Activer/désactiver les effets du nuanceur de pixels

Active ou désactive les effets du nuanceur de pixels activés dans le terminal. Si l'utilisateur a spécifié un nuanceur valide avec `experimental.pixelShaderPath`, cette action active/désactive ce nuanceur. Cela entraîne également le basculement de « l'effet façon rétro du terminal », qui est activé avec le paramètre de profil `experimental.retroTerminalEffect`.

Nom de la commande : `toggleShaderEffects`

Liaison par défaut :

JSON

```
{ "command": "toggleShaderEffects" }
```

> ⊗ **Attention**
>
> L'action `toggleRetroEffect` n'est plus disponible dans les versions 1,6 et ultérieures. Nous vous recommandons d'utiliser `toggleShaderEffects` à la place.

Définir le modèle de couleurs

Modifie le modèle de couleurs actif.

Nom de la commande : `setColorScheme`

Paramètres

Nom	Nécessité	Accepte	Description
`colorScheme`	Obligatoire	String	`name` du modèle de couleurs à appliquer.

Exemple de combinaison :

JSON

```
{ "command": { "action": "setColorScheme", "colorScheme": "Campbell" },
"keys": "" }
```

Ajouter une marque de défilement (fonctionnalité expérimentale)

Ajoute une marque de défilement à la mémoire tampon de texte. S'il y a déjà une sélection, la marque est placée au niveau de la sélection ; sinon, elle est placée sur la ligne du curseur. Il s'agit d'une fonctionnalité expérimentale et sa pérennité n'est pas garantie.

Nom de la commande : `addMark`

Paramètres

Nom	Nécessité	Accepte	Description
`color`	Facultatif	Chaîne au format hexadécimal : `"#rgb"` ou `"#rrggbb"`	Couleur de la marque.

Exemple de combinaison :

JSON

```
{ "command": { "action": "addMark", "color": "#ff00ff" } }
```

Fait défiler jusqu'à la marque (fonctionnalité expérimentale)

Fait défiler jusqu'à la marque de défilement dans la direction donnée. Il s'agit d'une fonctionnalité expérimentale et sa pérennité n'est pas garantie.

Nom de la commande : `scrollToMark`

Paramètres

Nom	Nécessité	Accepte	Description
`direction`	Obligatoire	`"first"`, `"previous"`, `"next"`, `"last"`	Sens dans lequel faire défiler.

Exemple de combinaison :

JSON

```
{ "command": { "action": "scrollToMark", "direction": "previous" } }
```

Effacer la marque (fonctionnalité expérimentale)

Efface la marque de défilement à la position actuelle, soit au niveau d'une sélection s'il y en a une, soit à la position du curseur. Il s'agit d'une fonctionnalité expérimentale et sa pérennité n'est pas garantie.

Nom de la commande : `clearMark`

Exemple de combinaison :

```JSON
{ "command": { "action": "clearMark" } }
```

Effacer toutes les marques (fonctionnalité expérimentale)

Efface toutes les marques de défilement dans la mémoire tampon de texte. Il s'agit d'une fonctionnalité expérimentale et sa pérennité n'est pas garantie.

Nom de la commande : `clearAllMarks`

Exemple de combinaison :

```JSON
{ "command": { "action": "clearAllMarks" } }
```

Exportation de mémoire tampon

Tampon d'exportation

Permet à l'utilisateur d'exporter le texte de la mémoire tampon dans un fichier. Si le fichier n'existe pas, il est créé. Si le fichier existe déjà, son contenu est remplacé par le texte de la mémoire tampon du terminal.

Nom de la commande : `exportBuffer`

Copier les liaisons par défaut :

```JSON
{ "command": { "action": "exportBuffer" } }
```

Paramètres

Nom	Nécessité	Accepte	Description
`path`	Facultatif	String	S'il est fourni, le terminal exporte le contenu de la mémoire tampon dans le fichier donné. Sinon, le terminal ouvre un sélecteur vous permettant de choisir le fichier vers lequel exporter le contenu.

Commandes globales

Appel global

Il s'agit d'une action spéciale qui fonctionne globalement dans le système d'exploitation, et non uniquement dans le contexte de la fenêtre du terminal. Quand vous appuyez dessus, cette action appelle la fenêtre du terminal. Les propriétés de cette action permettent de contrôler quelle fenêtre est appelée, où elle appelée et comment elle se comporte lors de son appel.

Remarques

- Les touches liées aux actions `globalSummon` dans le terminal ne fonctionnent pas dans d'autres applications quand le terminal est en cours d'exécution. La fenêtre du terminal a toujours le focus.
- Si une autre application en cours d'exécution utilise déjà les `keys` données avec l'API `RegisterHotKey`, le terminal ne peut pas écouter ces combinaisons de touches.
- Les instances avec et sans élévation de privilèges du terminal ne peuvent pas utiliser les mêmes touches. Il en va de même pour les versions Préversion et Stable du terminal. La première lancée l'emportera toujours.
- Ces combinaisons de touches ne fonctionnent que lorsqu'une instance du terminal est déjà en cours d'exécution. Pour lancer automatiquement le terminal dès la connexion, consultez startOnUserLogin.

Nom de la commande : `globalSummon`

Liaison par défaut :

Cette commande n'est pas actuellement liée dans les paramètres par défaut.

```
```

JSON

```
{ "keys": "", "command": { "action": "globalSummon" } }
```

Paramètres

Nom	Nécessité	Accepte	Description
`desktop`	Facultatif	`any`, `toCurrent`, `onCurrent`	Contrôle la manière dont le terminal doit interagir avec les bureaux virtuels. • `"any"` :laisse la fenêtre sur le bureau où elle se trouve déjà (bascule vers ce bureau lorsque la fenêtre est activée). • `"toCurrent"` (*valeur par défaut*) : déplace la fenêtre vers le bureau virtuel actuel. • `"onCurrent"` : appelle uniquement la fenêtre si elle se trouve déjà sur le bureau virtuel actuel.
`monitor`	Facultatif	`any`, `toCurrent`, `toMouse`	Contrôle le moniteur à partir duquel ou vers lequel la fenêtre sera appelée. • `"any"` :appelle la fenêtre la plus récemment utilisée, quel que soit le moniteur où elle se trouve actuellement. • `"toCurrent"` : appelle la fenêtre la plus récemment utilisée sur le moniteur avec la fenêtre de premier plan active. • `"toMouse"` (*valeur par défaut*) : appelle la fenêtre la plus récemment utilisée sur le moniteur où se trouve le curseur de la souris.
`name`	Facultatif	String	En cas d'omission (*valeur par défaut*), utilise `monitor` et `desktop` pour rechercher la fenêtre la plus récemment utilisée à appeler. Si cette action est fournie, appelle la fenêtre dont le nom ou l'ID correspond à la valeur `name` donnée. Si aucune fenêtre de ce type n'existe, crée une fenêtre portant ce nom.
`dropdownDuration`	Facultatif	Integer	La valeur par défaut est `0`. Si cette action est fournie avec un nombre positif, « fait glisser » la fenêtre depuis le haut de l'écran à l'aide d'une animation qui dure `dropdownDuration` millisecondes. `200` est une valeur raisonnable pour ce paramètre.

Nom	Nécessité	Accepte	Description
`toggleVisibility`	Facultatif	`true`, `false`	La valeur par défaut est `true`. Quand la valeur est `true`, le fait d'appuyer sur les touches affectées pour cette action ignore (réduit) la fenêtre quand celle-ci est actuellement la fenêtre de premier plan. Quand la valeur est `false`, le fait d'appuyer sur les touches affectées ne fait que mettre la fenêtre au premier plan.

Quand `name` est fourni *avec* `monitor` ou `desktop`, `name` se comporte comme suit :

- `desktop`
  - `"any"` : accède au bureau où la fenêtre donnée se trouve déjà.
  - `"toCurrent"` : si la fenêtre se trouve sur un autre bureau virtuel, elle est déplacée vers celui qui est actuellement actif.
  - `"onCurrent"` : si la fenêtre se trouve sur un autre bureau virtuel, elle est déplacée vers celui qui est actuellement actif.
- `monitor`
  - `"any"` : laisse la fenêtre sur le moniteur où elle se trouve déjà.
  - `"toCurrent"` : si la fenêtre se trouve sur un autre moniteur, elle est déplacée vers celui avec la fenêtre de premier plan active.
  - `"toMouse"` : si la fenêtre se trouve sur un autre moniteur, elle est déplacée vers celui où se trouve le curseur de la souris.

Les propriétés `desktop` et `monitor` peuvent être combinées des manières suivantes :

Combinaisons	`"desktop": "any"`	`"desktop": "toCurrent"`	`"desktop": "onCurrent"`	Non inclus
`"monitor": "any"`	Accède au bureau où se trouve la fenêtre (position inchangée)	Déplace la fenêtre vers ce bureau (position inchangée)	S'il n'y en a pas sur ce bureau : • En crée une à la position par défaut Sinon : • Active celle sur ce bureau (sans la déplacer)	Appeler la fenêtre MRU

Combinaisons	`"desktop": "any"`	`"desktop": "toCurrent"`	`"desktop": "onCurrent"`	**Non inclus**
`"monitor": "toCurrent"`	Accède au bureau où la fenêtre se trouve, passe au moniteur avec la fenêtre de premier plan	Déplace la fenêtre vers ce bureau, passe au moniteur avec la fenêtre de premier plan	S'il n'y en a pas sur ce bureau : • En crée une Sinon : • Active celle sur ce bureau, passe au moniteur avec la fenêtre de premier plan	Appelle la fenêtre MRU sur le moniteur avec la fenêtre de premier plan
`"monitor": "toMouse"`	Accède au bureau où la fenêtre se trouve, passe au moniteur avec la souris	Déplace la fenêtre vers ce bureau, passe au moniteur avec la souris	S'il n'y en a pas sur ce bureau : • En crée une Sinon : • Active celle sur ce bureau, passe au moniteur avec la souris	Appelle la fenêtre MRU sur le moniteur avec la souris
Non incluses	Conserve la position actuelle	Passe au bureau actuel	Sur le bureau actuel uniquement	N/A

Exemples

jsonc

```
// Summon the most recently used (MRU) window, to the current virtual
desktop,
// to the monitor the mouse cursor is on, without an animation. If the
window is
// already in the foreground, then minimize it.
{ "keys": "ctrl+1", "command": { "action": "globalSummon" } },
```

```
// Summon the MRU window, by going to the virtual desktop the window is
// currently on. Move the window to the monitor the mouse is on.
{ "keys": "ctrl+2", "command": { "action": "globalSummon", "desktop": "any"
} },

// Summon the MRU window to the current desktop, leaving the position of the
window untouched.
{ "keys": "ctrl+3", "command": { "action": "globalSummon", "monitor": "any"
} },

// Summon the MRU window, by going to the virtual desktop the window is
// currently on, leaving the position of the window untouched.
{ "keys": "ctrl+4", "command": { "action": "globalSummon", "desktop": "any",
"monitor": "any" } },

// Summon the MRU window with a dropdown duration of 200ms.
{ "keys": "ctrl+5", "command": { "action": "globalSummon",
"dropdownDuration": 200 } },

// Summon the MRU window. If the window is already in the foreground, do
nothing.
{ "keys": "ctrl+6", "command": { "action": "globalSummon",
"toggleVisibility": false } },

// Summon the window named "_quake". If no window with that name exists,
then create a new window.
{ "keys": "ctrl+7", "command": { "action": "globalSummon", "name": "_quake"
} }
```

Ouvrir la fenêtre en mode Quake

Cette action est une variante spéciale de l'action globalSummon. Elle appelle spécifiquement la fenêtre Quake. Il s'agit d'un raccourci pour l'action `globalSummon` suivante :

JSON

```
{
"keys": "win+`",
"command": {
"action": "globalSummon",
"name": "_quake",
"dropdownDuration": 200,
"toggleVisibility": true,
"monitor": "toMouse",
"desktop": "toCurrent"
}
}
```

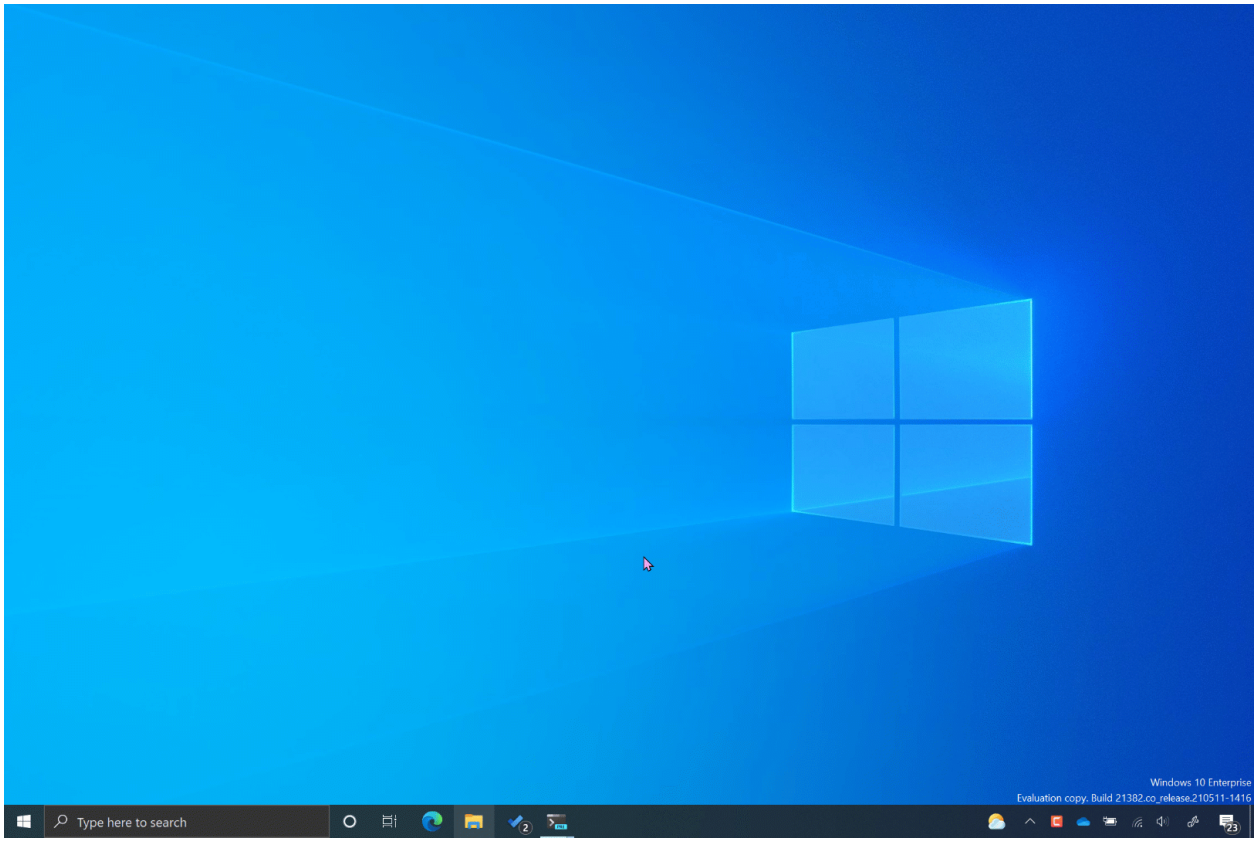
Si vous souhaitez modifier le comportement de l'action `quakeMode`, nous vous recommandons de créer une entrée `globalSummon` dans `actions` avec les paramètres de votre choix.

Nom de la commande : `quakeMode`

Liaison par défaut :

JSON

```
{ "keys": "win+`", "command": { "action": "quakeMode" } }
```



Exécuter plusieurs actions

Cette action permet à l'utilisateur de lier plusieurs actions séquentielles à une seule commande.

Nom de la commande : `multipleActions`

Paramètres

Nom	Nécessité	Accepte	Description

Nom	Nécessité	Accepte	Description
`actions`	Obligatoire	Tableau d'actions	Liste de `action` à exécuter.

Exemple

jsonc

```
{ "name": "Create My Layout", "command": {
    "action": "multipleActions",
    "actions": [
        // Create a new tab with 3 panes
        { "action": "newTab", "tabTitle": "Work", "colorScheme": "One Half
Dark" },
        { "action": "splitPane", "split": "vertical", "profile": "Windows
PowerShell", "tabTitle": "Work", "colorScheme": "Campbell Powershell", },
        { "action": "splitPane", "split": "horizontal", "profile": "Windows
PowerShell", "tabTitle": "Work", "colorScheme": "Campbell Powershell", },

        // Create a second tab
        { "action": "newTab", "tabTitle": "Misc"},

        // Go back to the first tab and zoom the first pane
        { "action": "prevTab", "tabSwitcherMode": "disabled" },
        { "action": "moveFocus", "direction": "first"},
        "togglePaneZoom"
        ]
}}
```

Dissocier les touches (désactiver les combinaisons de touches)

Vous pouvez désactiver les combinaisons de touches ou « dissocier » les touches associées à une commande. Cela peut s'avérer nécessaire lors de l'utilisation d'applications de terminal sous-jacentes (comme VIM). La touche dissociée est passée au terminal sous-jacent.

Nom de la commande : `unbound`

Exemple utilisant unbound :

Par exemple, pour dissocier les touches de raccourci `Alt`+`Maj`+`-` et `Alt`+`Maj`+`=`, incluez ces commandes dans la section `actions` de votre fichier settings.json.

JSON

```
{
    "actions": [
        { "command": "unbound", "keys": "alt+shift+-" },
        { "command": "unbound", "keys": "alt+shift+=" }
    ]
}
```

Exemple utilisant null :

Vous pouvez également dissocier une combinaison de touches associée par défaut à une action en définissant `"command"` avec la valeur `null`. Cela permet également d'associer la combinaison de touches au paramètre d'application de ligne de commande au lieu d'exécuter l'action par défaut.

JSON

```
{
   "command" : null, "keys" : ["ctrl+v"]
},
```

Scénario d'utilisation :

Le Terminal Windows utilise la combinaison de touches `Ctrl`+`V` pour la commande Coller. Quand vous utilisez une ligne de commande WSL, vous pouvez utiliser une application Linux telle que Vim pour modifier des fichiers. Cependant, Vim s'appuie sur la combinaison de touches `Ctrl`+`v` pour utiliser le mode visuel par blocs (blockwise Visual). Cette combinaison de touches sera bloquée et la commande Coller du Terminal Windows sera prioritaire, sauf si le paramètre `unbound` est ajusté dans votre fichier settings.json de manière à associer la combinaison de touches à l'application de ligne de commande Vim plutôt qu'à la commande du Terminal Windows.

Paramètres du profil général dans le Terminal Windows

Article • 17/03/2023

Les paramètres répertoriés ci-dessous sont spécifiques à chaque profil unique. Si vous souhaitez qu'un paramètre s'applique à tous vos profils, vous pouvez l'ajouter à la section `defaults` au-dessus de la liste des profils dans votre fichier settings.json.

JSON

```
"defaults":
{
    // SETTINGS TO APPLY TO ALL PROFILES
},
"list":
[
    // PROFILE OBJECTS
]
```

Nom

Il s'agit du nom du profil qui sera affiché dans le menu déroulant. Cette valeur est également utilisée comme « titre » à transmettre au shell au démarrage. Certains shells (comme `bash`) peuvent choisir d'ignorer cette valeur initiale, tandis que d'autres (`Command Prompt`, `PowerShell`) peuvent utiliser cette valeur pendant la durée de vie de l'application. Ce comportement de « titre » peut être substitué à l'aide de `tabTitle`.

Nom de la propriété : `name`

Nécessité : Obligatoire

Accepte : String

Ligne de commande

Il s'agit de l'exécutable utilisé dans le profil.

Nom de la propriété : `commandline`

Nécessité : Facultatif

Accepte : Nom de fichier exécutable sous forme de chaîne

Valeur par défaut : `"cmd.exe"`

Répertoire de départ

Il s'agit du répertoire dans lequel le shell démarre lorsqu'il est chargé.

Nom de la propriété : `startingDirectory`

Nécessité : Facultatif

Accepte : Emplacement du dossier sous forme de chaîne

Valeur par défaut : `"%USERPROFILE%"`

REMARQUE : Si le répertoire de démarrage n'est pas défini, la valeur par défaut est `"%USERPROFILE%"` (chemin relatif à vos paramètres utilisateur, par exemple `C:\Users\<your username>`). Toutefois, si le répertoire de départ est explicitement défini sur `null`, vous obtiendrez des résultats différents selon l'endroit à partir duquel vous lancez le terminal.

Exemple : Démarrez le profil PowerShell dans le dossier *GitHubRepos* de votre répertoire *Documents*. Pour cela, recherchez le profil powershell.exe et ajoutez `"startingDirectory": "%USERPROFILE%/Documents/GitHubRepos",`.

Exemple avec WSL : Quand vous définissez le répertoire de départ pour une distribution Linux installée avec WSL, utilisez le format `"startingDirectory": "\\\\wsl$\\DISTRO NAME\\home\\USERNAME"` en remplaçant les espaces réservés par les noms appropriés de votre distribution. Par exemple : `"startingDirectory": "\\\\wsl$\\Ubuntu-20.04\\home\\user1"`. Pour déclarer ce chemin si vous utilisez l'interface utilisateur des paramètres du Terminal Windows à la place du fichier settings.json, vous pouvez utiliser le bouton **Parcourir...** et sélectionner votre répertoire de départ ou entrer le chemin WSL comme suit : `//wsl.localhost/DISTRO NAME/home/USERNAME`. Par exemple : `//wsl.localhost/Ubuntu-20.04/home/user1`.

Comportement par défaut : Si la valeur startingDirectory n'est pas spécifiée, vous obtiendrez des résultats différents en fonction de l'emplacement de lancement du terminal :

- Si vous exécutez le Terminal Windows à partir du menu Démarrer : C:\windows\system32

- Si vous exécutez wt.exe à partir du menu Démarrer : C:\windows\system32
- Si vous exécutez wt.exe à partir de Win+R : %USERPROFILE%
- Si vous exécutez wt.exe à partir de la barre d'adresses de l'Explorateur : le dossier, quel qu'il soit, que vous consultiez.

> ⓘ **Notes**
>
> Les barres obliques inverses doivent être placées dans une séquence d'échappement. Par exemple, entrez `C:\Users\USERNAME\Documents` sous la forme `C:\\Users\\USERNAME\\Documents`.

Icône

Définit l'icône qui s'affiche dans l'onglet, le menu déroulant, la liste de raccourcis et le sélecteur d'onglet.

Nom de la propriété : `icon`

Nécessité : Facultatif

Accepte : Emplacement du fichier en tant que chaîne ou emoji

Exemple : En plaçant l'image d'icône `ubuntu.ico` dans le dossier situé dans `%LOCALAPPDATA%\Packages\Microsoft.WindowsTerminal_8wekyb3d8bbwe\RoamingState`, vous pouvez afficher l'icône en ajoutant cette ligne au profil dans votre fichier settings.json : `"icon": "ms-appdata:///roaming/ubuntu.ico"`.

Titre de l'onglet

Si cette valeur est définie, `name` est remplacé par le titre à passer au shell au démarrage. Certains shells (comme `bash`) peuvent choisir d'ignorer cette valeur initiale, tandis que d'autres (`Command Prompt`, `PowerShell`) peuvent utiliser cette valeur pendant la durée de vie de l'application. Si vous souhaitez savoir comment faire pour que le shell définisse votre titre, consultez le didacticiel sur les titres d'onglet.

Nom de la propriété : `tabTitle`

Nécessité : Facultatif

Accepte : String

Exécuter automatiquement en tant qu'administrateur

Si cette propriété est définie, ce profil s'ouvre automatiquement dans une fenêtre « avec élévation de privilèges » (exécution en tant qu'administrateur) par défaut. Si vous exécutez ce profil à partir d'une fenêtre sans élévation de privilèges, une nouvelle fenêtre de terminal avec élévation de privilèges est créée pour héberger ce profil. Si vous lancez ce profil à partir d'une fenêtre ayant déjà des privilèges élevés, il s'ouvre sous un nouvel onglet.

Quand cette propriété a la valeur `false`, l'ouverture de ce profil dans une fenêtre avec élévation de privilèges ne lance pas une fenêtre *sans élévation de privilèges* pour héberger ce profil. Le profil s'ouvre simplement dans la fenêtre avec élévation de privilèges (exécution en tant qu'administrateur).

Si vous définissez cette propriété dans `profiles.defaults`, *tous* les profils sont lancés par défaut en tant qu'administrateur, sauf si vous définissez spécifiquement la propriété avec la valeur false.

Cette propriété peut être substituée dans les actions newTab et splitPane, avec la propriété `elevate`.

Les onglets avec et sans élévation de privilèges ne peuvent pas se trouver dans la même fenêtre de terminal. Pour plus d'informations, consultez le FAQ.

Nom de la propriété : `elevate`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Masquer le profil dans la liste déroulante

Si `hidden` est défini sur `true`, le profil n'apparaîtra pas dans la liste des profils. Cela peut être utilisé pour masquer les profils par défaut et les profils générés de manière dynamique, tout en les laissant dans votre fichier de paramètres. Pour en savoir plus sur les profils dynamiques, consultez la page Profils dynamiques.

Nom de la propriété : `hidden`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Paramètres du profil d'apparence dans le Terminal Windows

Article • 17/03/2023

Les paramètres listés ci-dessous affectent les paramètres visuels de chaque profil séparément. Si vous souhaitez qu'un paramètre s'applique à tous vos profils, vous pouvez l'ajouter à la section `defaults` au-dessus de la liste des profils dans votre fichier settings.json.

JSON

```
"defaults":
{
    // SETTINGS TO APPLY TO ALL PROFILES
},
"list":
[
    // PROFILE OBJECTS
]
```

Texte

Modèle de couleurs

Il s'agit du nom du modèle de couleurs utilisé dans le profil. Les modèles de couleurs sont définis dans l'objet `schemes`. Vous trouverez des informations plus détaillées sur la page Modèles de couleurs.

En plus d'un nom de modèle de couleurs unique, cette propriété accepte une paire de noms de modèles de couleurs, comme ceci :

JSON

```
"colorScheme":
{
    "light": "One Half Light",
    "dark": "One Half Dark",
},
```

Si vous spécifiez la propriété de cette manière, le terminal bascule automatiquement entre les deux modèles de couleurs donnés en fonction du thème de l'application. Le terminal suit la propriété theme.applicationTheme du thème sélectionné du terminal. Si `applicationTheme` est défini sur `system`, le modèle de couleurs correspondant au thème du système d'exploitation est utilisé à la place.

Nom de la propriété : `colorScheme`

Nécessité : Facultatif

Accepte : le nom du modèle de couleurs sous la forme d'une chaîne ou d'un objet avec une propriété `light` ou `dark`

Valeur par défaut : `"Campbell"`

> ⓘ **Important**

> La spécification d'une paire de modèles de couleurs `light` et `dark` n'est disponible que dans la **préversion de Terminal Windows**.

Police

Il s'agit de la structure dans laquelle les autres paramètres de police doivent être définis. Vous pouvoir voir à quoi cela peut ressembler dans le fichier JSON ci-dessous.

Nom de la propriété : `font`

Nécessité : Facultatif

Font face

Il s'agit du nom du type de police utilisé dans le profil. Le terminal essaiera de revenir à Consolas s'il est introuvable ou non valide. Pour en savoir plus sur les autres variantes de la police par défaut, Cascadia Mono, accédez à la page Cascadia Code.

Nom de la propriété : `face` (défini dans l'objet `font`)

Nécessité : Facultatif

Accepte : Nom de police sous forme de chaîne

Valeur par défaut : `"Cascadia Mono"`

Taille de police

Définit la taille de police du profil en points.

Nom de la propriété : `size` (défini dans l'objet `font`)

Nécessité : Facultatif

Accepte : Entier

Valeur par défaut : `12`

Épaisseur de police

Cela définit l'épaisseur (légèreté ou lourdeur des traits) de la police du profil.

Nom de la propriété : `weight` (défini dans l'objet `font`)

Nécessité : Facultatif

Accepte : `"normal"`, `"thin"`, `"extra-light"`, `"light"`, `"semi-light"`, `"medium"`, `"semi-bold"`, `"bold"`, `"extra-bold"`, `"black"`, `"extra-black"` ou un entier correspondant à la représentation numérique de l'épaisseur de police OpenType

Valeur par défaut : `"normal"`

Exemple de police

JSON

```
"font": {
    "face": "Cascadia Mono",
    "size": 12,
    "weight": "normal"
}
```

Important

Cet objet `font` est uniquement disponible dans la **préversion du Terminal Windows** (version 1.10+). Avant cette version, vous devez utiliser les propriétés `fontFace`, `fontSize` et `fontWeight` séparément, comme ceci :

JSON

```
"fontFace": "Cascadia Mono",
"fontSize": 12,
"fontWeight": "normal"
```

Caractéristiques de police

Définit les caractéristiques de police OpenType pour la police donnée.

Nom de la propriété : `features` (défini dans l'objet `font`)

Nécessité : Facultatif

Accepte : Propriétés de la caractéristique au format `"string": integer`

Exemple :

jsonc

```
// Enables ss01 and disables ligatures
"font": {
    "face": "Cascadia Code",
    "features": {
        "ss01": 1,
        "liga": 0
    }
}
```

Axes de police

Définit les axes de police OpenType pour la police donnée.

Nom de la propriété : `axes` (défini dans l'objet `font`)

Nécessité : Facultatif

Accepte : Propriétés de l'axe au format `"string": integer`

Exemple :

jsonc

```
// Sets the font to italic
"font": {
    "face": "Cascadia Code",
    "axes": {
        "ital": 1
    }
}
```

Mise en forme du texte intense

Cette commande contrôle la mise en forme du texte « intense » dans le terminal. Le texte « intense » est mis en forme avec la séquence d'échappement `\x1b[1m`.

Nom de la propriété : `intenseTextStyle`

Nécessité : Facultatif

Accepte : `"none"`, `"bold"`, `"bright"`, `"all"`

- `"all"` : affiche le texte intense en **gras** et avec une couleur brillante
- `"bold"` : affiche le texte intense en **gras**, mais pas avec une couleur brillante
- `"bright"` : affiche le texte intense avec une couleur brillante, mais pas en gras
- `"none"` : le terminal ne fait rien de spécial pour le texte intense

Valeur par défaut : `"all"`

Effets façon rétro pour le terminal

Lorsque cette valeur est définie sur `true`, le terminal émule un affichage CRT classique avec des lignes de balayage et des bords de texte flous. Il s'agit d'une fonctionnalité expérimentale dont l'existence à long terme n'est pas garantie.

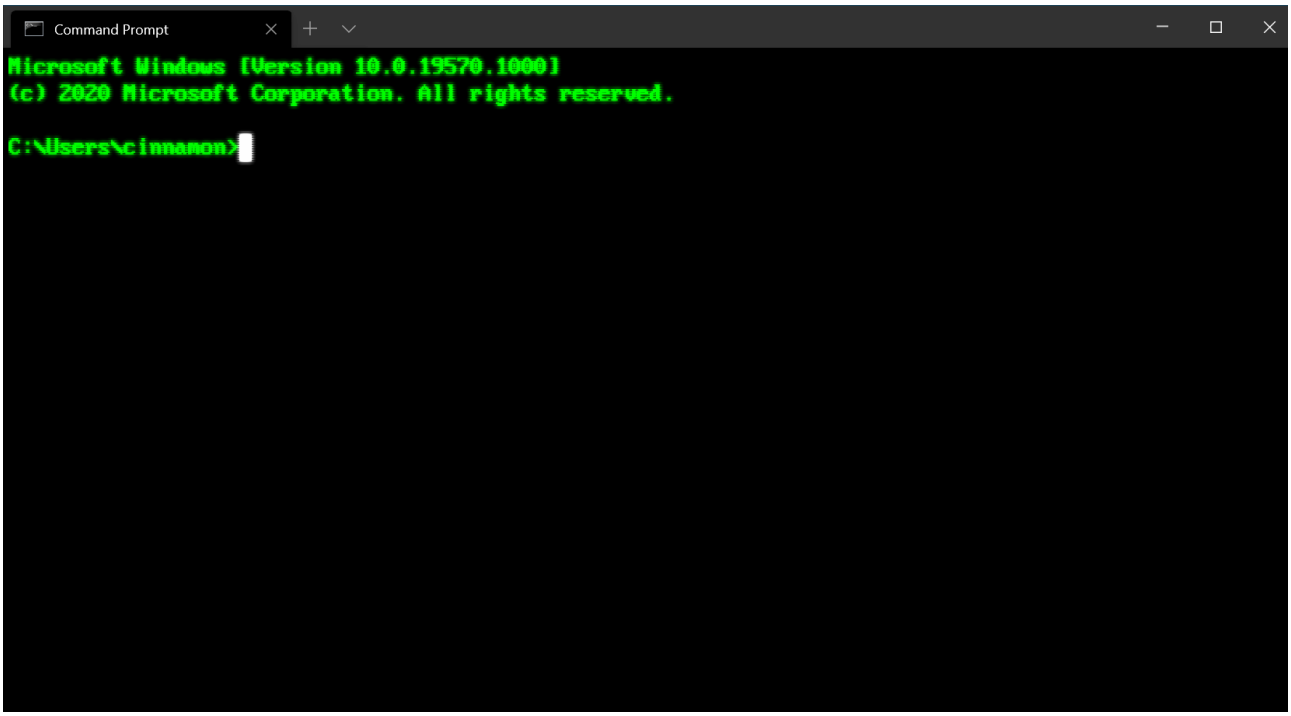
Ce paramètre est remplacé si `experimental.pixelShaderPath` est défini.

Nom de la propriété : `experimental.retroTerminalEffect`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`



Configuration: Invite de commandes façon rétro

Curseur

Forme de curseur

Définit la forme du curseur pour le profil. Les curseurs possibles sont les suivants : `"bar"` (▏), `"vintage"` (▬), `"underscore"` (_), `"filledBox"` (▉), `"emptyBox"` (▯), `"doubleUnderscore"` (‗)

Nom de la propriété : `cursorShape`

Nécessité : Facultatif

Accepte : `"bar"`, `"vintage"`, `"underscore"`, `"filledBox"`, `"emptyBox"`, `"doubleUnderscore"`

Valeur par défaut : `"bar"`

Hauteur du curseur

Définit la hauteur en pourcentage du curseur en partant du bas. Cela ne fonctionne que si `cursorShape` est défini sur `"vintage"`.

Nom de la propriété : `cursorHeight`

Nécessité : Facultatif

Accepte : Entier compris entre 1 et 100

Images d’arrière-plan et icônes

Le Terminal Windows vous permet de spécifier des images d'arrière-plan et des icônes personnalisées à l'aide de l'interface utilisateur des paramètres ou du fichier settings.json pour chacun de vos profils de ligne de commande. Vous pouvez ainsi configurer, personnaliser et styliser chacun de vos profils de manière indépendante. Pour cela, spécifiez votre `backgroundImage` préférée, positionnez-la avec `backgroundImageAlignment`, définissez son opacité avec `backgroundImageOpacity` et/ou spécifiez comment votre image remplira l'espace disponible avec `backgroundImageStretchMode`.

Par exemple :

JSON

```
    "backgroundImage": "C:\\Users\\username\\OneDrive\\WindowsTerminal\\bg-ubuntu-256.png",
    "backgroundImageAlignment": "bottomRight",
    "backgroundImageOpacity": 0.1,
    "backgroundImageStretchMode": "none"
```

Vous pouvez facilement appliquer votre collection d'images et d'icônes à l'ensemble de vos machines en stockant vos icônes et images dans OneDrive (comme indiqué ci-dessus).

Chemin de l'image d'arrière-plan

Définit l'emplacement de fichier de l'image à dessiner sur l'arrière-plan de la fenêtre. L'image d'arrière-plan peut être un fichier. jpg,. png ou. gif. `"desktopWallpaper"` définit définir le papier peint du Bureau comme image d'arrière-plan.

Nom de la propriété : `backgroundImage`

Nécessité : Facultatif

Accepte : Emplacement du fichier en tant que chaîne ou `"desktopWallpaper"`

Nous vous recommandons de stocker les images et icônes personnalisées dans des dossiers fournis par le système et de les référencer en utilisant les schémas d'URI appropriés. Les schémas d'URI permettent de référencer des fichiers indépendamment de leurs chemins physiques (qui peuvent être amenés à changer). Les schémas d'URI les plus utiles lors de la personnalisation d'images d'arrière-plan et d'icônes sont les suivants :

Schéma d'URI	Chemin physique correspondant	Utilisation/Description
`ms-appdata:///Local/`	`%localappdata%\Packages\Microsoft.WindowsTerminal_8wekyb3d8bbwe\LocalState\`	Fichiers par machine
`ms-appdata:///Roaming/`	`%localappdata%\Packages\Microsoft.WindowsTerminal_8wekyb3d8bbwe\RoamingState\`	Fichiers communs

> ⚠ **Avertissement**
>
> Ne vous fiez pas aux références de fichiers qui utilisent le schéma d'URI ms-appx (icônes). Ces fichiers sont considérés comme un détail d'implémentation interne, peuvent changer de nom ou d'emplacement ou pourront être omis à l'avenir.

Icônes

Le Terminal Windows affiche des icônes pour chaque profil généré par le terminal pour les shells intégrés, par exemple PowerShell Core, PowerShell et toute distribution Linux/WSL installée. Chaque profil fait référence à une icône d'une banque d'icônes par le biais du schéma d'URI ms-appx. Vous pouvez faire référence à vos propres icônes personnalisées en entrant un chemin dans votre fichier settings.json :

```JSON
    "icon" : "C:\\Users\\username\\OneDrive\\WindowsTerminal\\icon-ubuntu-32.png",
```

Les icônes doivent mesurer 32 x 32 px et être enregistrées dans un format d'image raster approprié (.PNG, .GIF, .ICO, etc.) pour éviter toute mise à l'échelle pendant l'exécution (provoquant un retard notable et une perte de qualité).

Si aucune icône n'est spécifiée pour une ligne de commande que vous avez installée, le Terminal Windows utilise par défaut ce glyphe à partir de la police Segoe Fluent :

Glyphe	Point de code Unicode	Description
	e756	CommandPrompt

Mode d'étirement de l'image d'arrière-plan

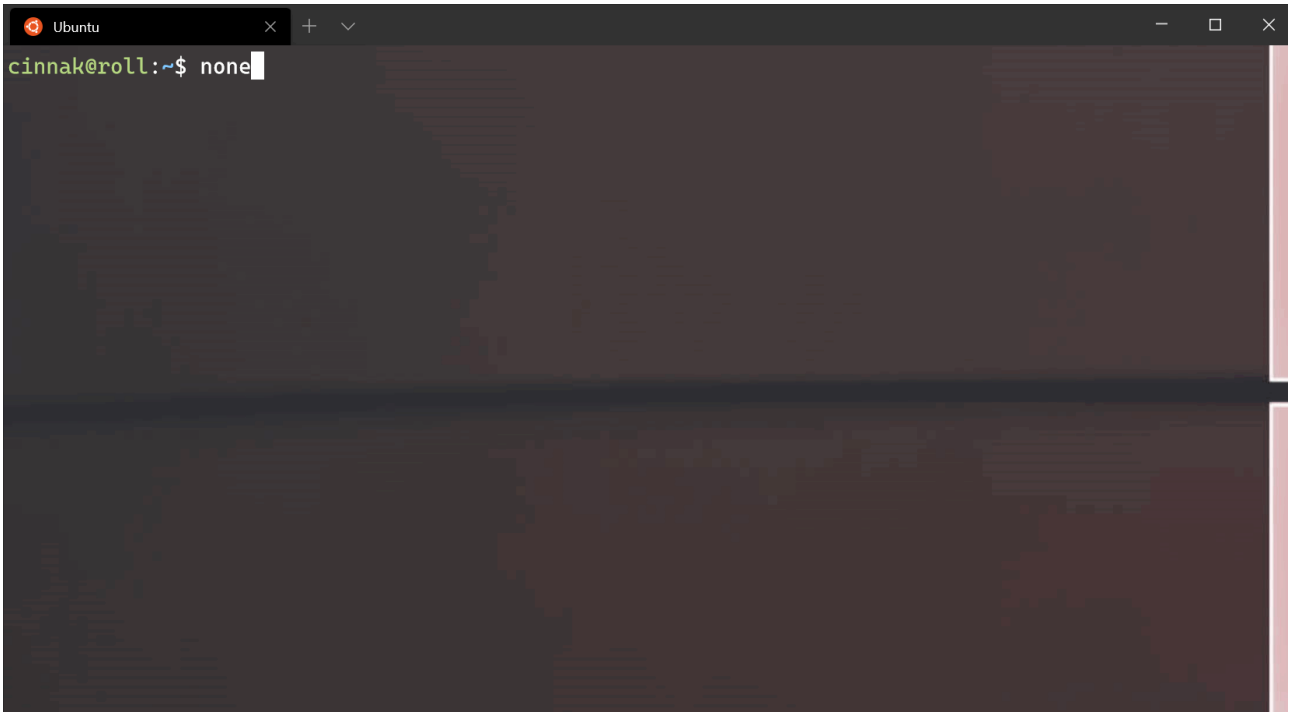
Définit la façon dont l'image d'arrière-plan est redimensionnée pour remplir la fenêtre.

Nom de la propriété : `backgroundImageStretchMode`

Nécessité : Facultatif

Accepte : `"none"`, `"fill"`, `"uniform"`, `"uniformToFill"`

Valeur par défaut : `"uniformToFill"`



Source de l'image d'arrière-plan

Alignement de l'image d'arrière-plan

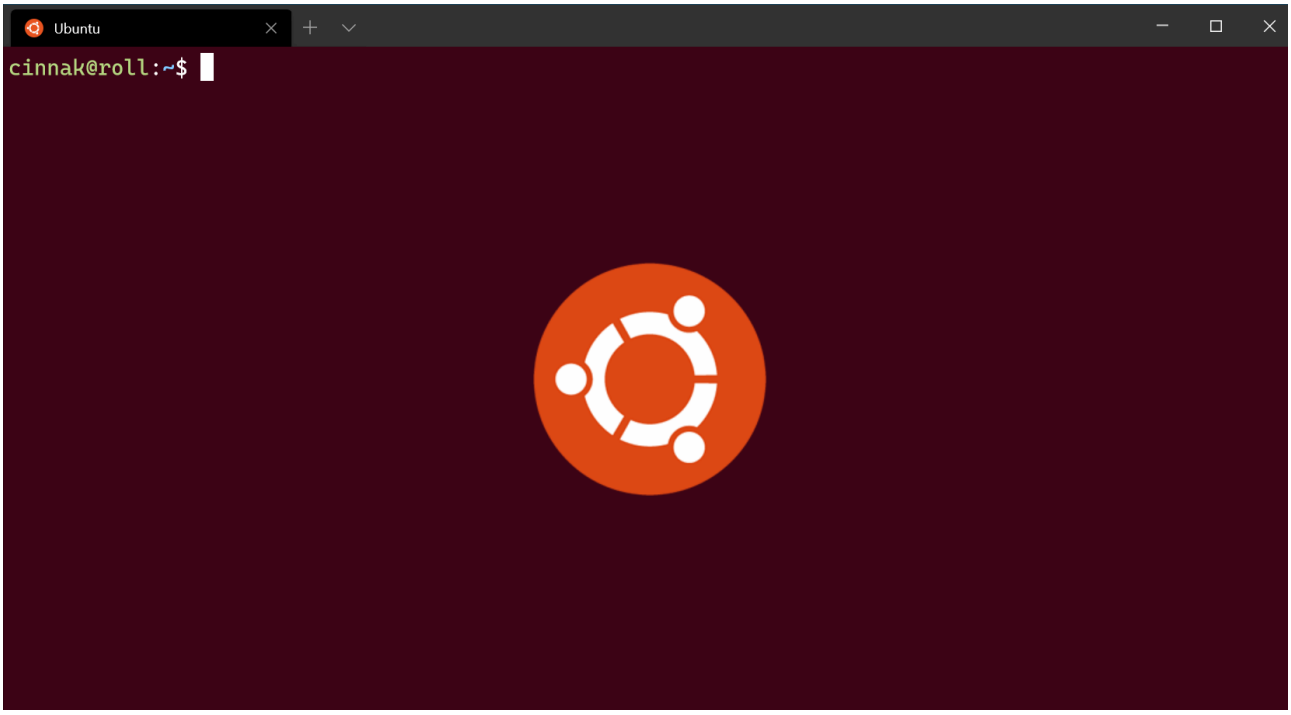
Définit la façon dont l'image d'arrière-plan s'aligne sur les limites de la fenêtre.

Nom de la propriété : `backgroundImageAlignment`

Nécessité : Facultatif

Accepte : `"center"`, `"left"`, `"top"`, `"right"`, `"bottom"`, `"topLeft"`, `"topRight"`, `"bottomLeft"`, `"bottomRight"`

Valeur par défaut : `"center"`



Source de l'image d'arrière-plan

Opacité de l'image d'arrière-plan

Définit la transparence de l'image d'arrière-plan.

Nom de la propriété : `backgroundImageOpacity`

Nécessité : Facultatif

Accepte : Nombre sous forme de valeur à virgule flottante comprise entre 0 et 1

Valeur par défaut : `1.0`

Transparence

Opacity

Définit la transparence de la fenêtre pour le profil. Accepte une valeur entière comprise entre 0 et 100 qui représente un « pourcentage d'opacité ». `100` est « entièrement opaque », `50` est semi-transparent et `0` est

entièrement transparent.

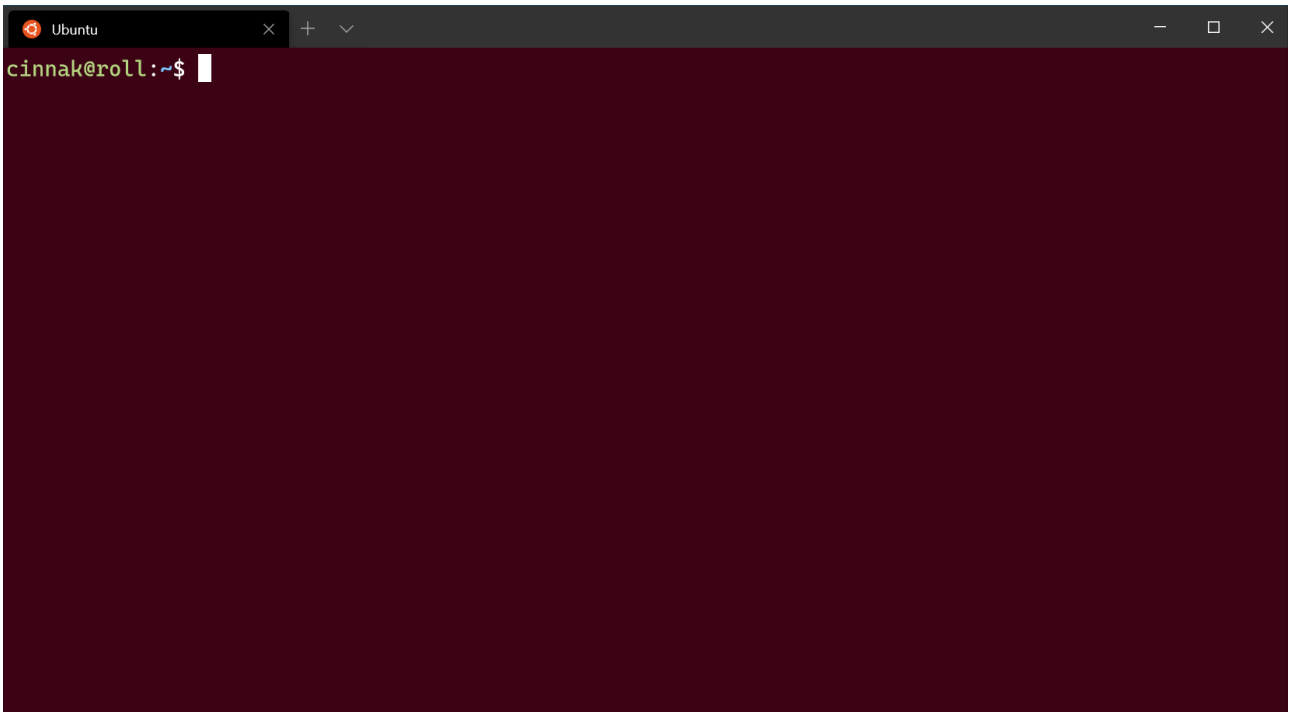
Quand `useAcrylic` est `true`, la fenêtre utilise le matériau acrylique pour créer un arrière-plan flou pour le terminal. Quand `useAcrylic` est false, le terminal utilise une opacité non floue.

Nom de la propriété : `opacity`

Nécessité : Facultatif

Accepte : Nombre sous forme d'une valeur entière comprise entre 0 et 100

Valeur par défaut : `100` quand `useAcrylic` est false, `50` quand `useAcrylic` est true.



> ⓘ **Important**
>
> Avant la version 1.12 du Terminal Windows, ce paramètre avait la valeur `acrylicOpacity` (float acceptant une valeur comprise entre 0.0 et 1.0 et défini par défaut sur 0,5), et l'opacité s'appliquait uniquement si `useAcrylic` était true. Sur 1.12+, `acrylicOpacity` continue de fonctionner normalement avec la valeur `opacity` équivalente.

> ⓘ **Important**
>
> L'opacité non floue (`"useAcrylic": false`) fonctionne uniquement sur Windows 11.

> ⓘ **Important**
>
> Quand Mica est activé dans les **paramètres du thème**, il apparaît sous le contenu du terminal quand le paramètre `opacity` du terminal est défini avec une valeur <100.

Activer l'acrylique

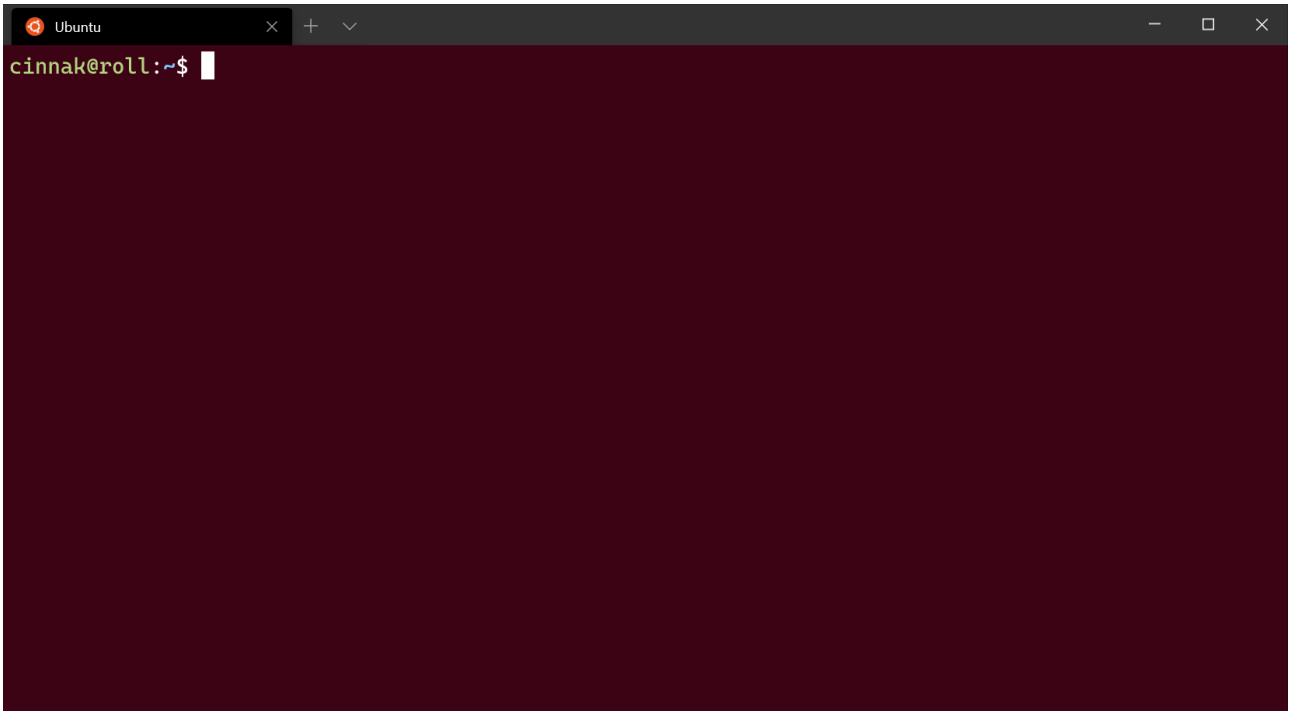
Lorsque cette valeur est définie sur `true`, la fenêtre aura un arrière-plan acrylique. Lorsqu'il est défini sur `false`, la fenêtre aura un arrière-plan ordinaire et sans texture. La transparence s'applique uniquement aux fenêtres ciblées en raison des limitations du système d'exploitation.

Nom de la propriété : `useAcrylic`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`



Fenêtre

Espacement

Définit l'espacement autour du texte dans la fenêtre. Ce paramètre accepte trois formats différents : `"#"` et `#` définissent le même espacement pour tous les côtés, `"#, #"` définit le même espacement pour les côtés gauche et droit, et `"#, #, #, #"` définit l'espacement individuellement pour les côtés gauche, haut, droit et bas.

Nom de la propriété : `padding`

Nécessité : Facultatif

Accepte : Valeurs en tant que chaîne au format `"#"`, `"#, #"` ou `"#, #, #, #"` ou valeur entière (`#`)

Valeur par défaut : `"8, 8, 8, 8"`

Visibilité de la barre de défilement

Définit la visibilité de la barre de défilement.

Nom de la propriété : `scrollbarState`

Nécessité : Facultatif

Accepte : `"visible"`, `"hidden"`, (à compter de la version 1.17, `"always"` sera inclus)

Paramètres de couleur

Couleur de l'onglet

Définit la couleur de l'onglet du profil. Elle est remplacée par une autre couleur si vous utilisez le sélecteur de couleurs d'onglet.

Nom de la propriété : `tabColor`

Nécessité : Facultatif

Accepte : Couleur sous forme de chaîne au format hexadécimal : `"#rgb"` ou `"#rrggbb"`

Couleur de premier plan

Modifie la couleur de premier plan du profil. Elle remplace la valeur `foreground` définie dans le modèle de couleurs si `colorScheme` est défini.

Nom de la propriété : `foreground`

Nécessité : Facultatif

Accepte : Couleur sous forme de chaîne au format hexadécimal : `"#rgb"` ou `"#rrggbb"`

Couleur d’arrière-plan

Modifie la couleur d’arrière-plan du profil avec ce paramètre. Elle remplace la valeur `background` définie dans le modèle de couleurs si `colorScheme` est défini.

Nom de la propriété : `background`

Nécessité : Facultatif

Accepte : Couleur sous forme de chaîne au format hexadécimal : `"#rgb"` ou `"#rrggbb"`

Couleur d’arrière-plan de sélection

Définit la couleur d’arrière-plan d’une sélection dans le profil. Elle remplace la valeur `selectionBackground` définie dans le modèle de couleurs si `colorScheme` est défini.

Nom de la propriété : `selectionBackground`

Nécessité : Facultatif

Accepte : Couleur sous forme de chaîne au format hexadécimal : `"#rgb"` ou `"#rrggbb"`

Ajuster les couleurs impossibles à distinguer

Ce paramètre ajuste la couleur du premier plan pour le rendre plus visible en fonction de la couleur d’arrière-plan. Quand il est défini sur `always`, les couleurs sont toujours ajustées. Quand il est défini sur `indexed`, les couleurs sont ajustées uniquement si elles font partie du jeu de couleurs. Quand il est défini sur `never`, les couleurs ne sont jamais ajustées.

Nom de la propriété : `adjustIndistinguishableColors`

Nécessité : Facultatif

Accepte : `always`, `indexed`, `never`

Couleur du curseur

Cette propriété définit la couleur du curseur du profil. Elle remplace la valeur `cursorColor` définie dans le modèle de couleurs si `colorScheme` est défini.

Nom de la propriété : `cursorColor`

Nécessité : Facultatif

Accepte : Couleur sous forme de chaîne au format hexadécimal : `"#rgb"` ou `"#rrggbb"`

Paramètres d’apparence inactifs

Objet que vous pouvez ajouter à un profil qui applique des paramètres au profil quand il n’a pas le focus. Ce paramètre accepte uniquement des paramètres d’apparence.

Nom de la propriété : `unfocusedAppearance`

Nécessité : Facultatif

Accepte : `backgroundImage`, `backgroundImageAlignment`, `backgroundImageOpacity`, `backgroundImageStretchMode`, `cursorHeight`, `cursorShape`, `cursorColor`, `colorScheme`, `foreground`, `background`, `selectionBackground`, `experimental.retroTerminalEffect`, `experimental.pixelShaderPath`

Exemple :

JSON

```
// Sets the profile's background image opacity to 0.3 when it is unfocused
"unfocusedAppearance":
{
    "backgroundImageOpacity": 0.3
},
```

Effets du nuanceur de pixels

Ce paramètre permet à un utilisateur de spécifier le chemin à un nuanceur de pixels personnalisé à utiliser avec le contenu du terminal. Il s'agit d'une fonctionnalité expérimentale dont l'existence à long terme n'est pas garantie. Pour plus d'informations sur la création de nuanceurs de pixels personnalisés pour le terminal, consultez cette documentation.

Si ce paramètre est défini, il remplace `experimental.retroTerminalEffect`.

Nom de la propriété : `experimental.pixelShaderPath`

Nécessité : Facultatif

Accepte : Chemin à un fichier de nuanceur `.hlsl`, en tant que chaîne

Paramètres du profil avancé dans le Terminal Windows

Article • 26/05/2023

Les paramètres répertoriés ci-dessous sont spécifiques à chaque profil unique. Si vous souhaitez qu'un paramètre s'applique à tous vos profils, vous pouvez l'ajouter à la section `defaults` au-dessus de la liste des profils dans votre fichier settings.json.

JSON

```
"defaults":
{
    // SETTINGS TO APPLY TO ALL PROFILES
},
"list":
[
    // PROFILE OBJECTS
]
```

Supprimer les modifications de titre

Lorsque cette option est définie sur `true`, `tabTitle` remplace le titre par défaut de l'onglet, et les messages de modification de titre de l'application sont supprimés. Si `tabTitle` n'est pas défini, `name` est utilisé à la place. Lorsque cette valeur est définie sur `false`, `tabTitle` se comporte normalement.

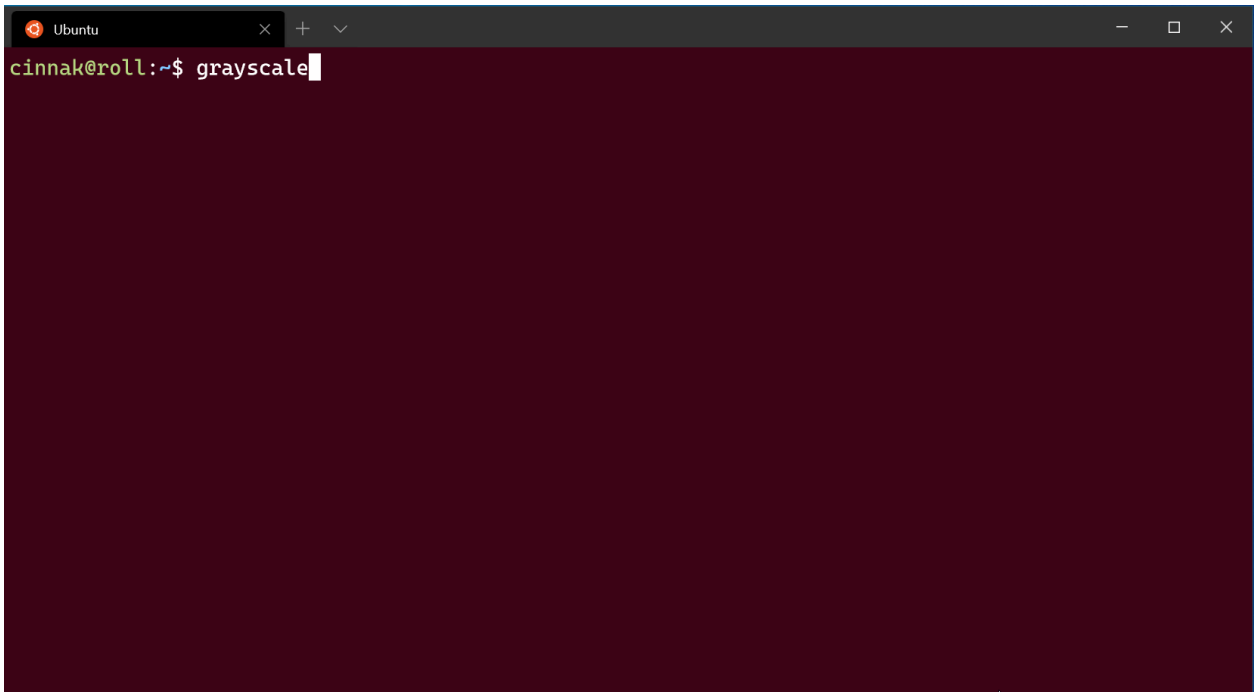
Nom de la propriété : `suppressApplicationTitle`

Nécessité : Facultatif

Accepte : `true`, `false`

Anticrénelage du texte

Contrôle le mode d'anticrénelage du texte dans le convertisseur. Notez que la modification de ce paramètre nécessite le démarrage d'une nouvelle instance du terminal.



Nom de la propriété : `antialiasingMode`

Nécessité : Facultatif

Accepte : `"grayscale"`, `"cleartype"`, `"aliased"`

Valeur par défaut : `"grayscale"`

Alias AltGr

Cela vous permet de contrôler si Terminal Windows traitera `Ctrl+Alt` comme alias pour `AltGr`.

Nom de la propriété : `altGrAliasing`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`

Faire défiler jusqu'à l'entrée lors de la saisie

Quand cette propriété est définie sur `true`, la fenêtre défile vers la ligne d'entrée de commande lors de la saisie. Lorsqu'elle est définie sur `false`, la fenêtre ne défile pas

lorsque vous commencez à saisir.

Nom de la propriété : `snapOnInput`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `true`

Taille de l'historique

Définit le nombre de lignes au-dessus de celles affichées dans la fenêtre vers lesquelles vous pouvez revenir. La taille maximale de l'historique est `32767`.

Nom de la propriété : `historySize`

Nécessité : Facultatif

Accepte : Entier

Valeur par défaut : `9001`

Comportement d'arrêt du profil

Définit la manière dont le profil réagit à l'arrêt ou l'échec du lancement. `"graceful"` ferme le profil lorsque `exit` est saisi ou lorsque le processus s'arrête normalement. `"always"` ferme toujours le profil et `"never"` ne ferme jamais le profil. `true` et `false` sont acceptés en tant que synonymes pour `"graceful"` et `"never"`, respectivement.

Nom de la propriété : `closeOnExit`

Nécessité : Facultatif

Accepte : `"graceful"`, `"always"`, `"never"`, `true`, `false`

Valeur par défaut : `"graceful"`

> ⓘ **Notes**

À l'invite de commandes Windows (cmd.exe), `exit` retourne le code de retour de la commande précédente. Si la commande que vous avez tapée avant `exit` a provoqué une erreur, `"closeOnExit": "graceful"` affiche toujours ce code d'erreur au lieu de fermer l'onglet.

Style de notification avec cloche

Contrôle ce qui se produit lorsque l'application émet un caractère BEL. Quand la valeur est `"all"`, le terminal joue un son et fait clignoter l'icône de la barre des tâches. Si le terminal n'a pas le focus, seule l'icône de la barre des tâches clignote.

Nom de la propriété : `bellStyle`

Nécessité : Facultatif

Accepte : `"all"`, `"audible"`, `"window"`, `"taskbar"`, `"none"`

Valeur par défaut : `"audible"`

Signal sonore

Lorsque `bellStyle` a la valeur `"all"` ou `"audible"`, cela vous permet de choisir le fichier audio pour la cloche. Si vous avez défini un groupe de sons, le terminal en choisit un au hasard.

Nom de la propriété : `bellSound`

Nécessité : Facultatif

Accepte : Emplacement du fichier sous la forme d'une chaîne ou d'un tableau d'emplacements de fichier sous forme de chaînes

---

Marques de défilement (préversion)

Les paramètres suivants modifient le comportement des marques de défilement dans le Terminal Windows.

Fonctionnalité expérimentale - Ajout automatique de marques de défilement (préversion)

Marque automatiquement les invites quand ce paramètre est défini sur `true`. Il s'agit d'une fonctionnalité expérimentale et sa pérennité n'est pas garantie.

Nom de la propriété : `experimental.autoMarkPrompts`

Nécessité : Facultatif

Accepte : `true`, `false`

> **Important**
>
> Cette fonctionnalité n'est disponible que dans la **préversion de Terminal Windows**.

Fonctionnalité expérimentale - Affichage des marques sur la barre de défilement (préversion)

Affiche les marques sur la barre de défilement quand ce paramètre est défini sur `true`. Il s'agit d'une fonctionnalité expérimentale et sa pérennité n'est pas garantie.

Nom de la propriété : `experimental.showMarksOnScrollbar`

Nécessité : Facultatif

Accepte : `true`, `false`

> **Important**
>
> Cette fonctionnalité n'est disponible que dans la **préversion de Terminal Windows**.

Moteur de rendu de texte expérimental

Permet d'utiliser le moteur de rendu de texte expérimental pour le profil. Il s'agit d'une fonctionnalité expérimentale dont l'existence à long terme n'est pas garantie. Une nouvelle instance du profil doit être ouverte pour que ce paramètre prenne effet.

Nom de la propriété : `experimental.useAtlasEngine`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Menu contextuel par clic droit (préversion)

Quand cette option est activée, un clic droit ouvre un menu contextuel comprenant des options de copier, coller, etc. Quand cette option est désactivée, un clic droit colle le contenu du Presse-papiers dans le terminal. Une fois l'intégration de l'interpréteur de commandes activée, un clic droit permet également de sélectionner la commande ou la sortie actuelle. Il s'agit d'une fonctionnalité expérimentale et sa pérennité n'est pas garantie.

Nom de la propriété : `experimental.rightClickContextMenu`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

> **Important**
>
> Cette fonctionnalité n'est disponible que dans la **préversion de Terminal Windows**.

Mode de pass-through VT

Lorsqu'il est défini sur vrai, ordonne au PTY de cette connexion d'utiliser le mode pass-through au lieu du moteur de simulation Conhost PTY d'origine. Il s'agit d'une fonctionnalité expérimentale et sa pérennité n'est pas garantie.

Nom de la propriété : `experimental.connection.passthroughMode`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

Identificateur unique

Les profils peuvent utiliser un GUID comme identificateur unique. Pour définir un profil comme profil par défaut, il devra disposer d'un GUID pour le paramètre global `defaultProfile`.

Nom de la propriété : `guid`

Nécessité : Obligatoire

Accepte : GUID sous forme de chaîne au format de Registre : `"{00000000-0000-0000-0000-000000000000}"`

> **Conseil**
>
> Vous pouvez exécuter `[guid]::NewGuid()` dans PowerShell pour générer un GUID pour votre profil personnalisé. Vous pouvez également utiliser le **générateur d'identificateurs GUID en ligne** ou, pour d'autres lignes de commande, le **générateur d'identificateurs UUID**.

Source

Stocke le nom du générateur de profils à l'origine du profil. *Aucune valeur détectable pour ce champ.* Pour plus d'informations sur les profils dynamiques, consultez la page Profils dynamiques.

Nom de la propriété : `source`

Nécessité : Facultatif

Accepte : String

> **Notes**
>
> Ce champ doit être omis lors de la déclaration d'un profil personnalisé. Il est utilisé par le terminal pour connecter des profils générés automatiquement à votre fichier de paramètres.

Paramètres de thème du Terminal Windows (préversion)

Article • 05/10/2023

Les paramètres indiqués ci-dessous affectent les visuels de la fenêtre de terminal proprement dite, et non l'apparence d'un onglet ni d'un volet individuel. Ces paramètres ne sont actuellement modifiables directement que dans le fichier settings.json. Ils ne sont pas configurables dans l'interface utilisateur des paramètres.

JSON

```
"theme": "dark"
"themes":
[
    // THEME OBJECTS
]
```

Pour voir des exemples de thèmes, consultez la galerie de thèmes.

Chacun des thèmes de la liste `themes` est constitué d'une collection d'objets de propriété, qui spécifient les propriétés des éléments individuels de l'application. Par exemple, le thème `"dark"` par défaut est le suivant :

JSON

```
{
    "name": "dark",
    "window": {
        "applicationTheme": "dark"
    },
    "tab": {
        "background": "terminalBackground",
        "unfocusedBackground": "#00000000"
    },
    "tabRow": {
        "unfocusedBackground": "#333333FF"
    }
},
```

Vous pouvez également configurer le terminal pour utiliser des thèmes distincts pour les modes Clair et Sombre dans le système d'exploitation et passer automatiquement d'un thème à l'autre quand le thème du système d'exploitation change. Pour cela, spécifiez la propriété `theme` en tant qu'objet contenant les clés `light` et `dark` :

JSON

```
"theme": { "dark": "<Dark Theme Name>", "light": "<Light Theme Name>" },
```

Nom du thème

Ce paramètre correspond au nom du thème. Chaque nom doit être unique. Les noms `dark`, `light` et `system` sont réservés aux thèmes par défaut intégrés.

Nom de la propriété : `name`

Nécessité : Obligatoire

Valeurs possibles : le nom du thème sous forme de chaîne

Fenêtre

Ces paramètres sont utilisés pour configurer l'apparence de l'ensemble de la fenêtre du terminal.

Nom de la propriété : `window`

Thème d'application

Définir le thème de l'interface utilisateur de l'application. Il permet de fixer le style des éléments d'interface utilisateur de l'application, notamment les boutons et la palette de commandes. Ce style peut être clair ou foncé. `"system"` utilise le même thème que Windows.

Nom de la propriété : `applicationTheme`

Nécessité : Facultatif

Accepte : `"system"`, `"dark"`, `"light"`

Valeur par défaut : `"dark"`

Mica

Cela active l'effet Mica sur cette fenêtre, sous toutes les autres couches de l'interface utilisateur. Pour que Mica soit visible, les couches situées au-dessus doivent être

transparentes. Par exemple, pour avoir une rangée d'onglets avec Mica, vous devez configurer le canal alpha de l'arrière-plan sur `0`, comme ceci :

JSON

```
{
    "name": "My Mica Theme",
    "tab":
    {
        "background": "terminalBackground"
    },
    "tabRow":
    {
        "background": "#00000000"
    },
    "window":
    {
        "applicationTheme": "system",
        "useMica": true
    }
},
```

Notez que lorsque Mica est activé pour la fenêtre, il est activé dans l'intégralité de la fenêtre, y compris comme toile de fond pour les volets du terminal dans la fenêtre. Cela signifie que les profils qui utilisent `opacity` sans `useAcrylic` activé feront apparaître le nouvel arrière-plan Mica. Il n'est actuellement pas possible d'avoir simultanément un arrière-plan transparent non flou pour le terminal et un arrière-plan Mica pour la rangée d'onglets.

Nom de la propriété : `useMica`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

> ⓘ Notes
>
> Mica n'est disponible que sur les builds Windows >= 22621.

Bordure de fenêtre

Cette option permet de définir la couleur de la bordure de fenêtre lorsque celle-ci est active. Lorsqu'elle est définie sur `null`, la bordure utilise la couleur par défaut pour le

thème du système d'exploitation.

Nom de la propriété : `frame`

Nécessité : Facultatif

Valeur possible : une couleur de thème

Valeur par défaut : `null`

> **Notes**
>
> Les couleurs de bordure de fenêtre sont uniquement disponibles sur Windows 11.

> **Important**
>
> Cette fonctionnalité n'est disponible que dans la **préversion de Terminal Windows**.

Bordure de fenêtre inactive

Cette option permet de définir la couleur de la bordure de fenêtre lorsque celle-ci est inactive. Lorsqu'elle est définie sur `null`, la bordure utilise la couleur par défaut pour le thème du système d'exploitation.

Nom de la propriété : `unfocusedFrame`

Nécessité : Facultatif

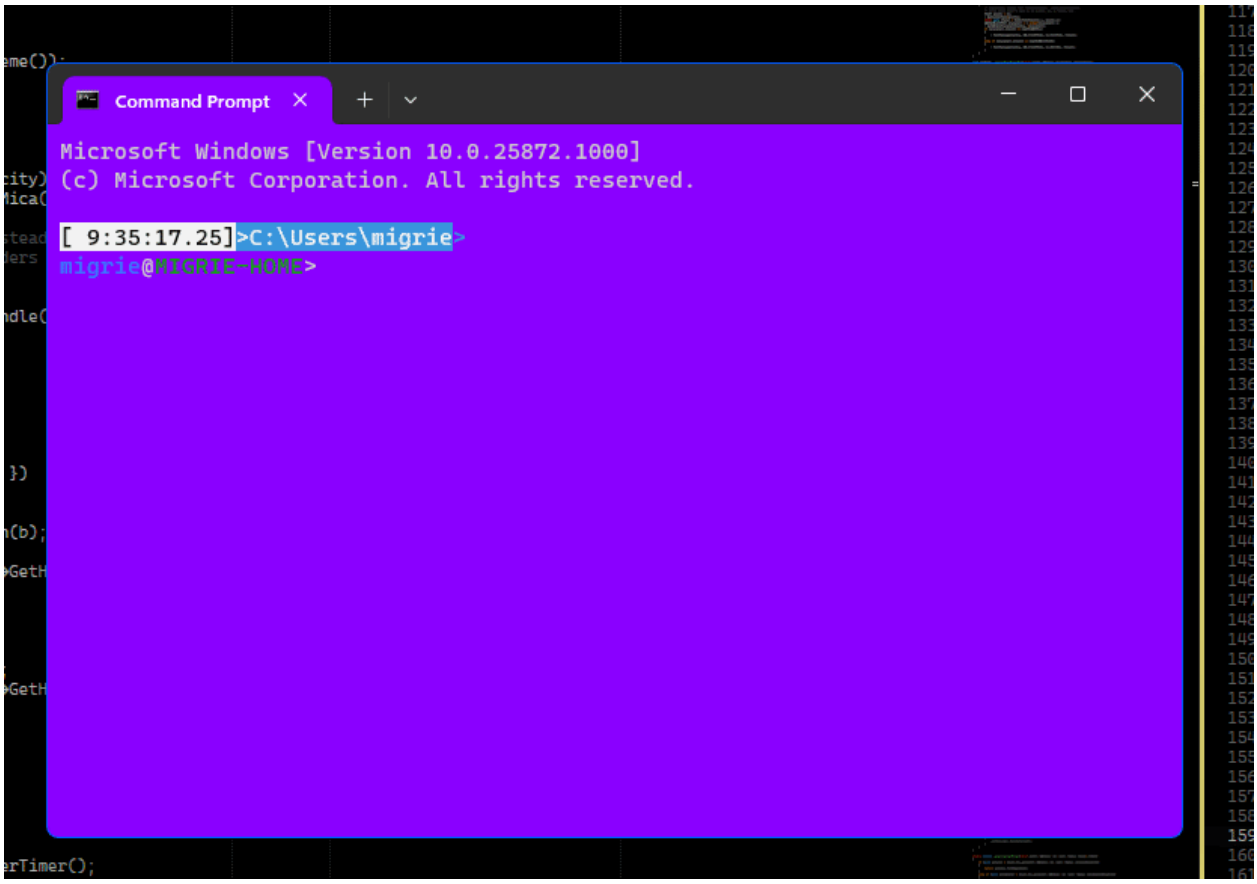
Valeur possible : une couleur de thème

Valeur par défaut : `null`

> **Important**
>
> Cette fonctionnalité n'est disponible que dans la **préversion de Terminal Windows**.

Version expérimentale : bordure de fenêtre arc-en-ciel

Lorsqu'il est activé, ce paramètre entraîne le cycle de la bordure de fenêtre via les couleurs de l'arc-en-ciel. Il s'agit d'une fonctionnalité expérimentale et sa pérennité n'est pas garantie. Lorsque ce paramètre est activé, il est prioritaire sur `frame` et `unfocusedFrame`.



Nom de la propriété : `experimental.rainbowFrame`

Nécessité : Facultatif

Accepte : `true`, `false`

Valeur par défaut : `false`

> **ⓘ Important**
>
> Cette fonctionnalité n'est disponible que dans la **préversion de Terminal Windows**.

Ligne d'onglets

Configurer l'apparence de la ligne d'onglets. Quand `showTabsInTitlebar` prend la valeur `true` (par défaut), ils configurent la barre de titre.

Nom de la propriété : `tabRow`

Couleur d'arrière-plan

Couleur de la ligne d'onglets lorsque la fenêtre se trouve au premier plan.

Nom de la propriété : `background`

Nécessité : Facultatif

Valeurs possibles : une couleur de thème.

Couleur de l'arrière-plan inactif

Couleur de la ligne d'onglets lorsque la fenêtre est inactive.

Nom de la propriété : `unfocusedBackground`

Nécessité : Facultatif

Valeurs possibles : une couleur de thème.

Tabulations

Il s'agit des paramètres qui contrôlent l'apparence des onglets individuels dans le terminal.

Nom de la propriété : `tab`

Couleur d'arrière-plan

Couleur de l'onglet actif. Cette valeur est remplacée si une valeur `tabColor` est définie dans un profil. La couleur est également écrasée lorsqu'une couleur est paramétrée à l'exécution avec le sélecteur de couleurs d'onglet.

Cette couleur est toujours traitée comme une couleur unie, même si elle est définie sur le `terminalBackground` d'un volet avec arrière-plan acrylique.

Nom de la propriété : `background`

Nécessité : Facultatif

Valeurs possibles : une couleur de thème.

Couleur de l'arrière-plan inactif

Couleur des onglets inactifs. Cette valeur est remplacée si une valeur `tabColor` est définie dans un profil. La couleur est également écrasée lorsqu'une couleur est paramétrée à l'exécution avec le sélecteur de couleurs d'onglet.

Cette couleur est toujours traitée comme une couleur unie, même si elle est définie sur le `terminalBackground` d'un volet avec arrière-plan acrylique.

Lorsque cette valeur est définie sur `terminalBackground` ou `accent`, elle utilise automatiquement une valeur alpha de 30 % pour être semi-transparente.

Nom de la propriété : `unfocusedBackground`

Nécessité : Facultatif

Valeurs possibles : une couleur de thème.

Affichage du bouton Fermer

Configure la façon dont le bouton « Fermer » de l'onglet apparaît. Ce paramètre accepte les valeurs suivantes :

- `"always"` : toujours afficher le bouton Fermer des onglets.
- `"hover"` : afficher le bouton Fermer sur l'onglet actif et tous les onglets sur lesquels passe la souris.
- `"never"` : ne jamais afficher le bouton Fermer des onglets. Cette valeur a également pour effet de désactiver la possibilité de fermer l'onglet avec le bouton central de la souris.
- `"activeOnly"` : afficher le bouton Fermer sur l'onglet actif uniquement.

Nom de la propriété : `showCloseButton`

Nécessité : Facultatif

Accepte : `"always"`, `"hover"`, `"never"`, `"activeOnly"`

Valeur par défaut : `"always"`

Couleurs du thème

Les couleurs utilisées dans les thèmes acceptent les valeurs RVBA, ainsi que quelques chaînes spéciales pour les valeurs personnalisées. Les valeurs acceptées sont les suivantes :

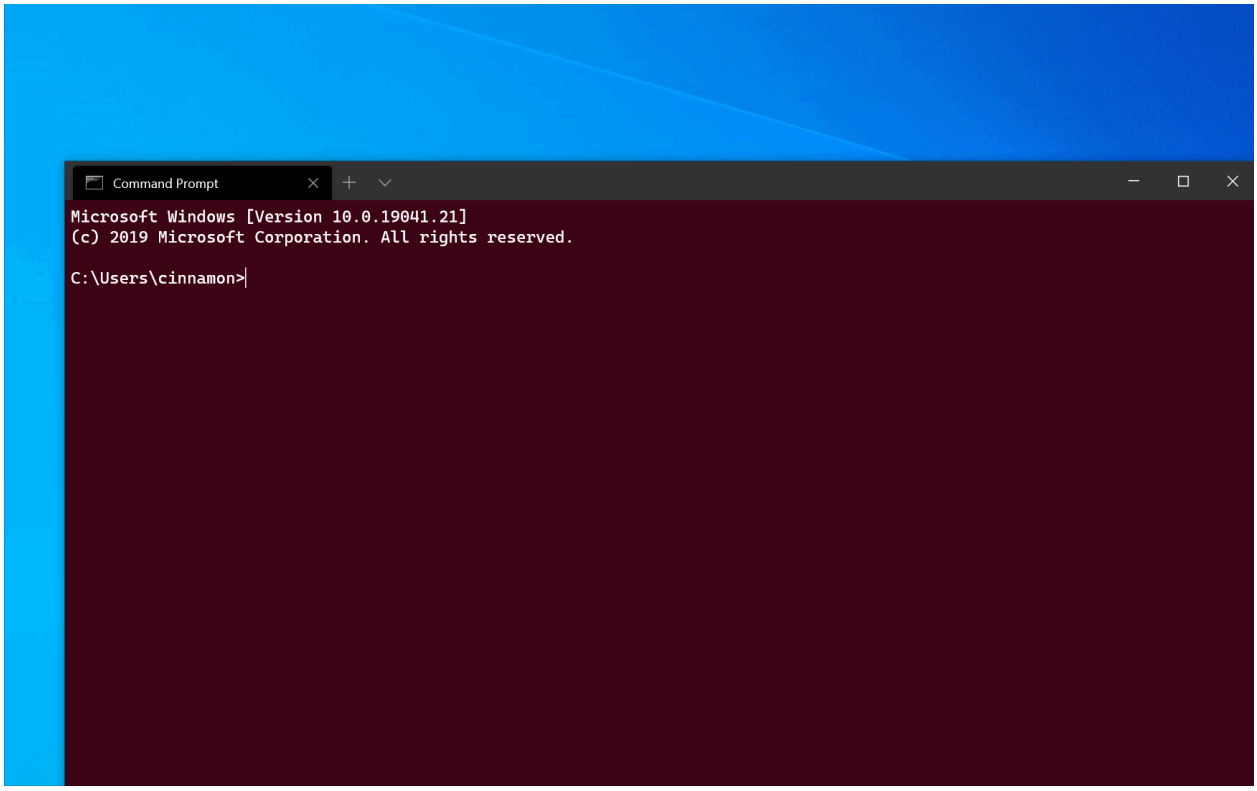
- `"#rgb`, `"#rrggbb` et `"#rrggbbaa` : une valeur de couleur RVB. Lorsque le canal alpha est omis, ces couleurs correspondent par défaut à un canal alpha entièrement opaque.
- `"accent"` : une valeur spéciale signifiant « la couleur d'accentuation définie dans les paramètres système ».
- `"terminalBackground"` : une valeur spéciale évaluée comme signifiant « la couleur d'arrière-plan du volet de terminal actif ». S'il existe plusieurs volets dans un onglet, il s'agit de la couleur du volet actif. Cette valeur utilise toujours le profil `background` et ignore tout ce qui provient d'une `backgroundImage` (le cas échéant).

Utilisation des arguments de ligne de commande pour le Terminal Windows

Article • 23/05/2023

Vous pouvez utiliser `wt.exe` pour ouvrir une nouvelle instance de Terminal Windows à partir de la ligne de commande. Vous pouvez également utiliser l'alias d'exécution `wt` à la place.

> **Notes**
>
> Si vous avez généré Terminal Windows à partir du code source sur **GitHub**, vous pouvez ouvrir cette build à l'aide de `wtd.exe` ou de `wtd`.



Syntaxe de la ligne de commande

La ligne de commande `wt` accepte deux types de valeurs : **options** et **commandes**. Les **options** constituent une liste d'indicateurs et autres paramètres qui peuvent contrôler le comportement de la ligne de commande `wt` dans son ensemble. Les **commandes** fournissent l'action ou la liste des actions séparées par des points-virgules, qui doivent être implémentées. Si aucune commande n'est spécifiée, la commande est supposée être `new-tab` par défaut.

Invite de commandes Windows

```
wt [options] [command ; ]
```

> ⓘ **Notes**
>
> Le comportement de la commande `wt.exe` peut être affecté par la **propriété windowingBehavior**. Ce paramètre peut être ajusté par défaut entre l'ouverture d'une nouvelle fenêtre ou l'ouverture d'un nouvel onglet.

Pour afficher un message d'aide répertoriant les arguments de ligne de commande disponibles, entrez : `wt -h`, `wt --help`, `wt -?` ou `wt /?`.

Options et commandes

Vous trouverez ci-dessous la liste complète des commandes et options prises en charge pour la ligne de commande `wt`.

Option	Description
`--help, -h, -?, /?`	Affiche le message d'aide.
`--maximized, -M`	Lance le terminal agrandi.
`--fullscreen, -F`	Lance le terminal en plein écran.
`--focus, -f`	Lance le terminal en mode Focus. Peut être combiné avec `maximized`.
`--pos x,y`	Lance le terminal à la position donnée. Vous pouvez omettre `x` ou `y` pour utiliser la valeur par défaut des paramètres.
`--window, -w window-id`	Exécute la commande donnée dans une fenêtre spécifique.

Le paramètre `--window` peut être utilisé pour envoyer des commandes à des fenêtres de terminal existantes.

`window-id` peut correspondre à l'ID entier d'une fenêtre ou au nom d'une fenêtre. Il accepte également les valeurs réservées suivantes :

- `new` ou `-1` : toujours exécuter cette commande dans une nouvelle fenêtre

- `last` ou `0` : toujours exécuter cette commande dans la dernière fenêtre utilisée

Si aucune fenêtre ne correspond au `window-id` donné, une fenêtre est créée avec cet ID ou ce nom.

Par exemple, la commande `wt -w _quake` ouvre une nouvelle « fenêtre en mode Quake » Si vous réexécutez cette commande, un nouvel onglet s'ouvre dans la fenêtre existante en mode Quake.

Commande new-tab

Permet de créer un onglet. Voir aussi l'action newTab.

Commande	Paramètre	Description	Valeurs
`new-tab`, `nt`	`--profile, -p profile-name`	Crée un onglet en fonction du nom de profil affecté.	Nom de profil
`new-tab`, `nt`	`--startingDirectory, -d starting-directory`	Crée un onglet en fonction du chemin du répertoire de départ affecté.	Chemin du répertoire
`new-tab`, `nt`	`commandline`	Crée un onglet en fonction de la ligne de commande affectée.	Exécutable avec des commandes facultatives
`new-tab`, `nt`	`--title`	Crée un onglet avec le titre affecté.	Texte à utiliser comme titre de l'onglet
`new-tab`, `nt`	`--tabColor`	Crée un onglet avec la couleur d'onglet affectée.	Couleur hexadécimale #RGB ou #RRGGBB
`new-tab`, `nt`	`--suppressApplicationTitle`	Remplace le paramètre `suppressApplicationTilte` du profil et lui affecte la valeur `true`.	
`new-tab`, `nt`	`--useApplicationTitle`	Remplace le paramètre `suppressApplicationTilte` du profil et lui affecte la valeur `false`.	
`new-tab`, `nt`	`--colorScheme scheme-name`	Remplace le paramètre `colorScheme` du profil et lui affecte le modèle nommé `scheme-name` dans les paramètres.	Nom d'un modèle de couleurs dans les paramètres

> **Conseil**
>
> Si vous modifiez le titre d'un onglet dans le Terminal Windows et que vous souhaitez conserver ce titre, vous devez activer l'option **suppressApplicationTitle** en lui affectant la valeur `true`.

Commande split-pane

Permet de créer un volet divisé. Voir aussi l'action splitPane.

Commande	Paramètre	Description	Valeurs
`split-pane`, `sp`	`-H, --horizontal`, `-V, --vertical`	Crée un volet de fenêtre divisé horizontalement ou verticalement.	N/A. Aucune valeur supplémentaire à affecter.
`split-pane`, `sp`	`--profile, -p profile-name`	Crée un volet de fenêtre divisé en fonction du profil de ligne de commande affecté. Si ce paramètre n'est pas affecté, le profil par défaut est utilisé.	Nom de profil
`split-pane`, `sp`	`--startingDirectory, -d starting-directory`	Crée un volet de fenêtre divisé en fonction du chemin du répertoire de départ affecté. Si ce paramètre n'est pas affecté, le répertoire de départ par défaut est utilisé.	Chemin du répertoire
`split-pane`, `sp`	`--title`	Crée un volet de fenêtre divisé avec le titre affecté.	Texte à utiliser comme titre de l'onglet
`split-pane`, `sp`	`--tabColor`	Crée un volet de fenêtre divisé avec la couleur d'onglet affectée.	Couleur hexadécimale #RGB ou #RRGGBB
`split-pane`, `sp`	`--size, -s size`	Crée un volet de fenêtre divisé avec la taille affectée.	Float qui spécifie la partie du volet parent à utiliser représentée par un décimal. Par exemple, `.4` pour représenter 40 % du volet parent.

Commande	Paramètre	Description	Valeurs
`split-pane`, `sp`	`commandline`	Crée un volet de fenêtre divisé basé sur le profil de ligne de commande affecté.	Exécutable avec des commandes facultatives
`split-pane`, `sp`	`--duplicate, -D`	Crée un volet de fenêtre divisé qui est un doublon du volet actuel.	N/A. Aucune valeur supplémentaire à affecter.
`split-pane`, `sp`	`--suppressApplicationTitle`	Remplace le paramètre `suppressApplicationTilte` du profil et lui affecte la valeur `true`.	
`split-pane`, `sp`	`--useApplicationTitle`	Remplace le paramètre `suppressApplicationTilte` du profil et lui affecte la valeur `false`.	
`split-pane`, `sp`	`--colorScheme scheme-name`	Remplace le paramètre `colorScheme` du profil et lui affecte le modèle nommé `scheme-name` dans les paramètres.	Nom d'un modèle de couleurs dans les paramètres

Commande focus-tab

Permet de déplacer le focus sur un onglet spécifique de la fenêtre. Voir aussi l'action switchToTab.

Commande	Paramètre	Description	Valeurs
`focus-tab`, `ft`	`--target, -t tab-index`	Met le focus sur un onglet spécifique en fonction du numéro d'index de l'onglet.	Index d'onglet au format entier

Commande move-focus

Permet de déplacer le focus dans la fenêtre. Voir aussi l'action moveFocus.

Commande	Paramètre	Description	Valeurs
`move-focus`, `mf`	`<direction>`	Déplace le focus entre les volets.	Voir les valeurs `direction` acceptées ci-dessous.

Valeurs `direction` acceptées

- `up`, `down`, `left` ou `right` : déplace le focus dans la direction donnée.
- `first` : déplace le focus sur le premier volet de nœud terminal dans l'arborescence.
- `previous` : déplace le focus sur le dernier volet utilisé avant le volet actif.
- `nextInOrder` ou `previousInOrder` : déplace le focus sur le volet suivant ou précédent selon l'ordre de création.

Commande move-pane

Permet de déplacer un volet dans la fenêtre. Voir aussi l'action movePane.

Commande	Paramètre	Description	Valeurs
`move-pane`, `mp`	`--tab,-t <index>`	Déplace le volet actif sur l'onglet donné dans la fenêtre.	Index indexé à zéro de l'onglet vers lequel déplacer le volet.

Commande swap-pane

Permet de permuter la position de deux volets dans la fenêtre. Voir aussi l'action swapPane.

Commande	Paramètre	Description	Valeurs
`swap-pane`	`<direction>`	Permute le volet avec le volet situé dans la direction donnée.	Voir les valeurs `direction` acceptées ci-dessous.

Valeurs `direction` acceptées (valeurs identiques à celles de la sous-commande `move-focus`)

- `up`, `down`, `left` ou `right` : permute le volet actif avec le volet situé dans la direction donnée.
- `first` : permute le volet actif avec le premier volet de nœud terminal de l'arborescence.
- `previous` : permute le volet actif avec le dernier volet utilisé avant le volet actif.
- `nextInOrder` ou `previousInOrder` : permute le volet actif avec le volet suivant ou précédent selon l'ordre de création.

Exemples d'argument de ligne de commande

Les commandes peuvent varier légèrement en fonction de la ligne de commande que vous utilisez.

Passage d'un argument au shell par défaut

Pour démarrer une instance du Terminal Windows et lui faire exécuter une commande, appelez `wt.exe` suivi de votre commande.

Voici un exemple montrant comment appeler le Terminal Windows pour passer un argument de commande ping et retourner une adresse IP :

PowerShell

```
wt ping learn.microsoft.com
```

Voici un exemple montrant comment appeler le Terminal Windows pour ouvrir un nouvel onglet avec une ligne de commande PowerShell, confirmer l'appel de la commande Start-Service et ouvrir un autre onglet avec l'invite de commandes Windows ouverte sur le répertoire `/k` :

Invite de commandes Windows

```
wt new-tab PowerShell -c Start-Service ; new-tab cmd /k dir
```

Cibler une fenêtre spécifique

Vous trouverez ci-dessous des exemples montrant comment cibler des fenêtres spécifiques avec l'option `--window,-w`.

Invite de commandes

Invite de commandes Windows

```
// Open a new tab with the default profile in the current window
wt -w 0 nt

// Open a new tab in a new window with the default profile
wt -w -1 nt

// Open a new tab in the first-created terminal window with the default
profile
wt -w 1 nt

// Open a new tab in the terminal window named foo with the default
profile. If foo does not exist, create a new window named foo.
wt -w foo nt
```

Ouvrir une nouvelle instance de profil

Pour ouvrir une nouvelle instance de terminal, dans le cas présent, la commande ouvre le profil nommé « Ubuntu-18,04 », entrez :

Invite de commandes

Invite de commandes Windows

```
wt -p "Ubuntu-18.04"
```

L'indicateur `-p` est utilisé pour spécifier quel profil de Terminal Windows ouvrir. Remplacez « Ubuntu-18.04 » par le nom d'un profil terminal que vous avez installé. Cela ouvrira toujours une nouvelle fenêtre. Le Terminal Windows ne peut pas encore ouvrir de nouveaux onglets ou volets dans une instance existante.

Cibler un répertoire

Pour spécifier le dossier qui doit être utilisé comme répertoire de démarrage de la console, dans le cas présent, le répertoire d:/, entrez :

Invite de commandes

Invite de commandes Windows

```
wt -d d:\
```

Plusieurs onglets

Pour ouvrir une nouvelle instance de terminal avec plusieurs onglets, entrez :

Invite de commandes

Invite de commandes Windows

```
wt ; ;
```

Pour ouvrir une nouvelle instance de terminal avec plusieurs onglets, dans le cas présent un profil d'invite de commandes et un profil PowerShell, entrez :

Invite de commandes

Invite de commandes Windows

```
wt -p "Command Prompt" ; new-tab -p "Windows PowerShell"
```

Plusieurs volets

Pour ouvrir une nouvelle instance de terminal avec un onglet contenant trois volets exécutant un profil d'invite de commandes, un profil PowerShell et votre profil par défaut exécutant une ligne de commande WSL, entrez :

Invite de commandes

Invite de commandes Windows

```
wt -p "Command Prompt" ; split-pane -p "Windows PowerShell" ; split-pane
-H wsl.exe
```

L'indicateur `-H` (ou `--horizontal`) indique que vous souhaitez que les volets soient fractionnés horizontalement. L'indicateur `-V` (ou `--vertical`) indique que vous souhaitez que les volets soient fractionnés verticalement.

Onglets et volets multiples

Les commandes `new-tab` et `split-pane` peuvent être séquencées pour recevoir plusieurs onglets, chacun ayant des volets divisés. Pour ouvrir une nouvelle instance de terminal avec deux onglets, chaque onglet ayant un répertoire différent et comprenant deux volets qui exécutent une invite de commandes et une ligne de commande WSL, entrez :

Invite de commandes

Invite de commandes Windows

```
wt -p "Command Prompt" ; split-pane -V wsl.exe ; new-tab -d c:\ ; split-
pane -H -d c:\ wsl.exe
```

Titre du volet

Pour ouvrir une nouvelle instance de terminal avec des titres personnalisés pour chaque volet de terminal, utilisez l'argument `--title`. Pour définir le titre de chaque volet lors de l'ouverture de plusieurs onglets, saisissez :

Invite de commandes

Invite de commandes Windows

```
wt --title tabname1 ; new-tab -p "Ubuntu-18.04" --title tabname2
```

Les volets d'un même onglet peuvent avoir des titres différents, qui se refléteront sur le titre de l'onglet en fonction du volet sélectionné. Pour nommer des volets indépendants, vous pouvez définir le titre après avoir divisé les volets en saisissant :

Invite de commandes

Invite de commandes Windows

```
wt --title pane1 ; split-pane -p "Command Prompt" --title pane2
```

Utilisation du titre de l'application

Pour ouvrir une nouvelle instance de terminal permettant aux applications qui s'y trouvent de définir le titre de l'onglet en envoyant des messages de changement de titre, utilisez l'indicateur `--useApplicationTitle`. Pour supprimer ces messages, utilisez l'indicateur `--suppressApplicationTitle`. Si aucun de ces indicateurs n'est fourni, le comportement est hérité des paramètres du profil. Pour ouvrir un onglet avec un titre `tabname` qui ne sera pas remplacé par l'application, entrez :

Invite de commandes

Invite de commandes Windows

```
wt --title tabname --suppressApplicationTitle
```

Couleur de l'onglet

Pour ouvrir une nouvelle instance de terminal avec des couleurs d'onglet personnalisées, utilisez l'argument `--tabColor`. Cet argument remplace la valeur définie dans le profil, mais peut également être substituée à l'aide du sélecteur de couleurs d'onglet. Dans l'exemple suivant, un terminal est créé avec deux onglets de couleurs différentes :

Invite de commandes

Invite de commandes Windows

```
wt --tabColor #009999 ; new-tab --tabColor #f59218
```

Quand l'argument `--tabColor` est défini pour un onglet, il est associé au premier volet de cet onglet. Dans un onglet à plusieurs volets, la couleur n'est donc appliquée que si le premier volet a le focus. Pour définir la couleur d'onglet pour d'autres volets, vous devez également ajouter le paramètre `--tabColor` à la sous-commande `split-pane`. Dans l'exemple ci-dessous, un onglet avec deux volets est créé avec les couleurs d'onglet spécifiées pour chaque volet :

PowerShell

```
wt new-tab --tabColor '#009999' `; split-pane --tabColor '#f59218'
```

Modèle de couleurs

Pour ouvrir une nouvelle instance de terminal avec un modèle de couleurs spécifique (au lieu du `colorScheme` défini dans le profil), utilisez l'argument `--colorScheme`. Cet argument remplace la valeur définie dans le profil.

Invite de commandes

Invite de commandes Windows

```
wt --colorScheme Vintage ; split-pane --colorScheme "Tango Light"
```

Focus de tabulation

Pour ouvrir une nouvelle instance de terminal avec un onglet spécifique en focus, utilisez l'indicateur `-t` (ou `--target`), ainsi que le numéro d'index de tabulation. Pour

ouvrir votre profil par défaut dans le premier onglet et dans le profil « Ubuntu-18,04 » mis en focus dans le deuxième onglet (`-t 1`), entrez :

Invite de commandes

Invite de commandes Windows

```
wt ; new-tab -p "Ubuntu-18.04" ; focus-tab -t 1
```

Exemples de commandes multiples à partir de PowerShell

Terminal Windows utilise le point-virgule `;` comme délimiteur pour séparer les commandes dans la ligne de commande `wt`. Malheureusement, PowerShell utilise également `;` comme séparateur de commandes. Pour contourner ce problème, vous pouvez utiliser les astuces suivantes pour exécuter plusieurs commandes `wt` à partir de PowerShell. Dans tous les exemples suivants, une nouvelle fenêtre de terminal est créée avec trois volets : l'un exécutant l'invite de commandes, l'autre s'exécutant avec PowerShell et le dernier exécutant WSL.

Les exemples suivants n'utilisent pas `start` pour exécuter la ligne de commande. Au lieu de cela, il existe deux autres méthodes d'échappement de la ligne de commande :

- Il suffit de traiter littéralement les points-virgules afin que `PowerShell` les ignore et les transmettent directement à `wt`.
- Si vous utilisez `--%`, PowerShell traitera le reste de la ligne de commande comme des arguments pour l'application.

PowerShell

```
wt new-tab "cmd" `; split-pane -p "Windows PowerShell" `; split-pane -H
wsl.exe
```

PowerShell

```
wt --% new-tab cmd ; split-pane -p "Windows PowerShell" ; split-pane -H
wsl.exe
```

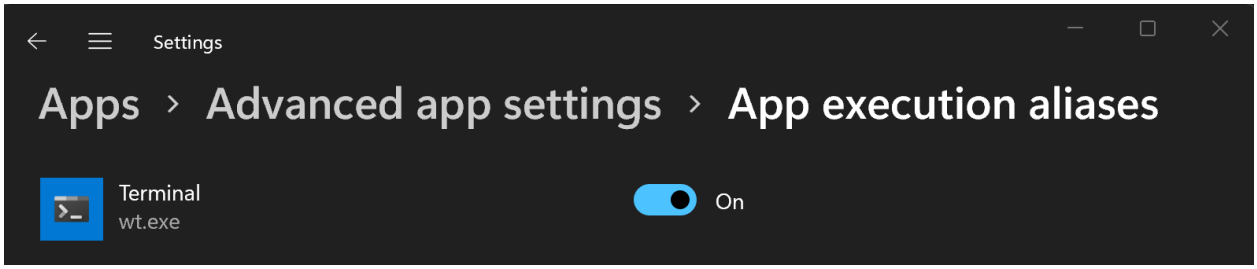
Dans ces deux exemples, la fenêtre de Terminal Windows nouvellement créée crée la fenêtre en analysant correctement tous les arguments de ligne de commande fournis.

Toutefois, ces méthodes ne sont *pas* actuellement recommandées, car PowerShell attend la fermeture de la fenêtre de terminal nouvellement créée avant de restituer le contrôle à PowerShell. Par défaut, PowerShell attendra toujours que les applications du Windows Store (comme Terminal Windows) se ferment avant de revenir à l'invite. Notez que cela est différent du comportement de l'invite de commandes, qui retourne immédiatement à l'invite.

Ajoutez l'exécutable Windows Terminal à votre chemin

Pour ajouter le fichier exécutable Terminal Windows (wt.exe) à votre chemin, activez son « alias d'exécution d'application » dans la page **Gérer les alias d'exécution d'application** dans Paramètres Windows. L'alias Terminal Windows est activé par défaut, mais il peut être utile de le confirmer si vous rencontrez des problèmes pour y accéder.

Si vous avez toujours des difficultés à accéder aux alias d'exécution d'application, vérifiez si votre chemin contient `%LOCALAPPDATA%\Microsoft\WindowsApps`. N'essayez pas de modifier `C:\Program Files\WindowsApps`.



Guide pratique pour utiliser la palette de commandes dans le Terminal Windows

Article • 21/03/2023

La palette de commandes vous permet de voir les actions que vous pouvez exécuter dans le Terminal Windows. Pour plus d'informations sur la façon dont les actions sont définies, consultez la page Actions.

Appel de la palette de commandes

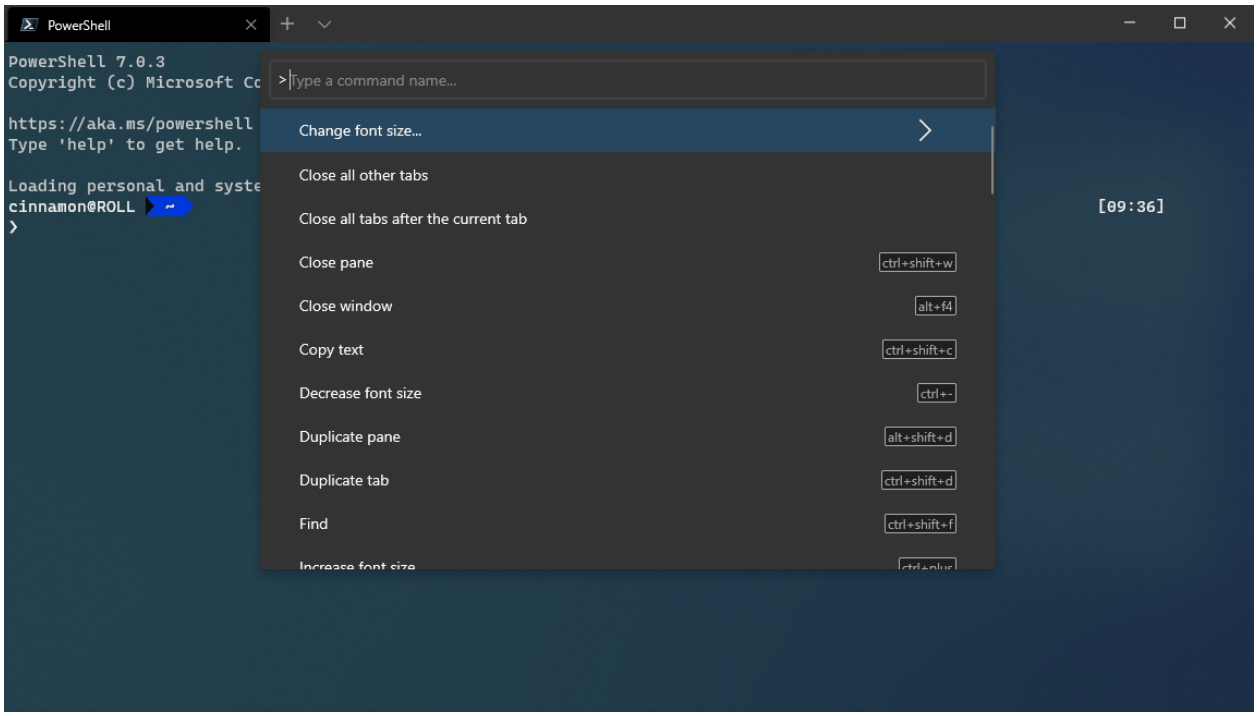
Vous pouvez appeler la palette de commandes en utilisant `Ctrl`+`Maj`+`P`. Pour personnaliser ceci, ajoutez la commande `commandPalette` à votre combinaisons de touches.

JSON

```
{ "command": "commandPalette", "keys": "ctrl+shift+p" }
```

Mode de ligne de commande

Pour entrer une commande `wt` dans la palette de commandes, supprimez le caractère `>` dans la zone de texte. La commande `wt` est alors exécutée dans la fenêtre active. Pour plus d'informations sur les commandes , consultez la wtpage Arguments de ligne de commande.



Vous pouvez ajouter une combinaison de touches personnalisée pour appeler la palette de commandes directement dans le mode de ligne de commande.

JSON

```
{ "command": "commandPalette", "launchMode": "commandLine", "keys": "" }
```

Ajout d'une icône à une commande

Vous pouvez ajouter une icône à une commande définie dans votre fichier settings.json qui apparaît dans la palette de commandes. Pour ce faire, ajoutez la propriété `icon` à l'action. Les icônes peuvent être un chemin à une image, un symbole de Segoe MDL2 Assets ou tout autre caractère, notamment des emojis.

JSON

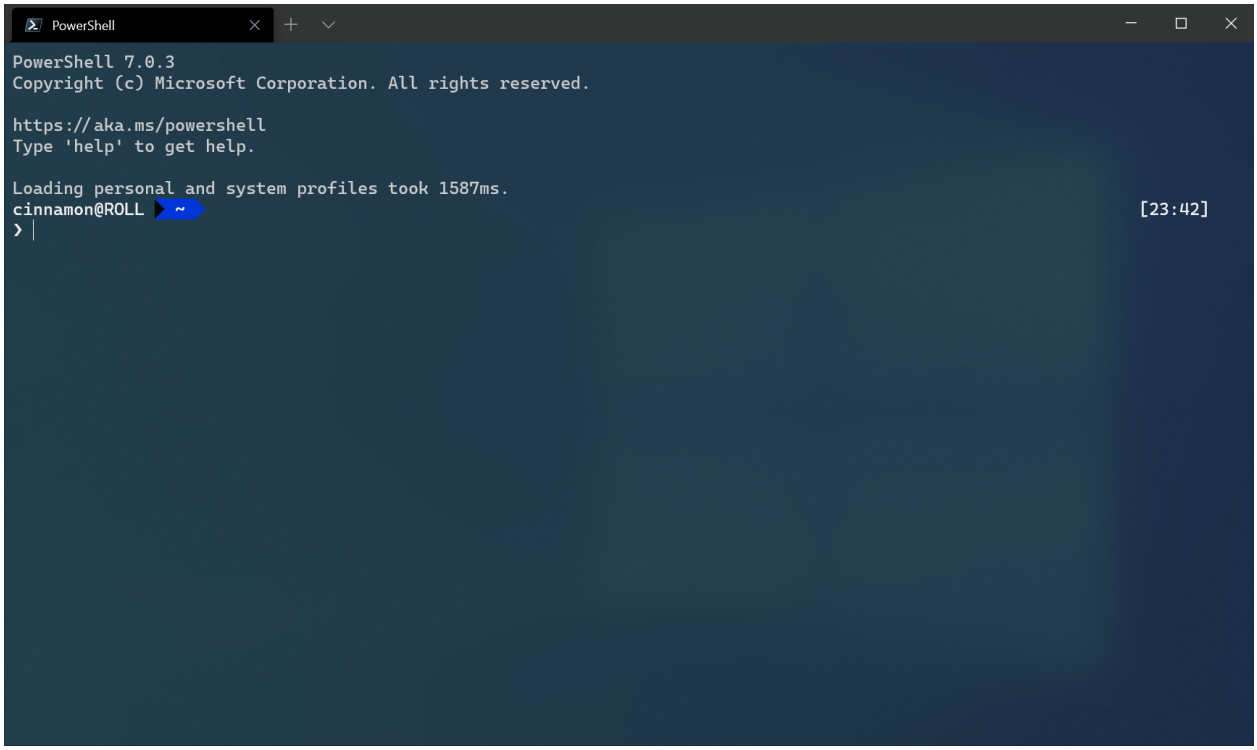
```
{ "icon": "C:\\Images\\my-icon.png", "name": "New tab", "command": "newTab",
"keys": "ctrl+shift+t" },
{ "icon": "\uE756", "name": "New tab", "command": "newTab", "keys":
"ctrl+shift+t" },
{ "icon": "⚡", "name": "New tab", "command": "newTab", "keys":
"ctrl+shift+t" }
```

Commandes imbriquées

Les commandes imbriquées vous permettent de regrouper plusieurs commandes sous un seul élément dans la palette de commandes. L'exemple ci-dessous regroupe les commandes de redimensionnement de police sous un élément de palette de commandes appelé « Change font size... ».

JSON

```
{
    "name": "Change font size...",
    "commands": [
        { "command": { "action": "adjustFontSize", "delta": 1 } },
        { "command": { "action": "adjustFontSize", "delta": -1 } },
        { "command": "resetFontSize" },
    ]
}
```



Commandes itérables

Les commandes itérables vous permettent de créer plusieurs commandes à la fois, générées à partir d'autres objets définis dans vos paramètres. Vous pouvez actuellement créer des commandes itérables pour vos profils et modèles de couleurs. Au moment de l'exécution, ces commandes sont développées en une seule commande pour chacun des objets du type donné.

Vous pouvez actuellement itérer au sein des propriétés suivantes :

`iterateOn`	**Propriété**	**Syntaxe des propriétés**

`iterateOn`	Propriété	Syntaxe des propriétés
`profiles`	`name`	`"name": "${profile.name}"`
`profiles`	`icon`	`"icon": "${profile.icon}"`
`schemes`	`name`	`"name": "${scheme.name}"`

Exemple

Créez une nouvelle commande d'onglet pour chaque profil.

JSON

```
{
    "iterateOn": "profiles",
    "icon": "${profile.icon}",
    "name": "${profile.name}",
    "command": { "action": "newTab", "profile": "${profile.name}" }
}
```

Dans l'exemple ci-dessus :

- `"iterateOn": "profiles"` génère une commande pour chaque profil.
- Au moment de l'exécution, le terminal remplace `${profile.icon}` par l'icône de chaque profil et `${profile.name}` par le nom de chaque profil.

Si vous avez trois profils :

JSON

```
"profiles": [
    { "name": "Command Prompt", "icon": null },
    { "name": "PowerShell", "icon": "C:\\path\\to\\icon.png" },
    { "name": "Ubuntu", "icon": null },
]
```

La commande ci-dessus se comporte comme les trois commandes suivantes :

JSON

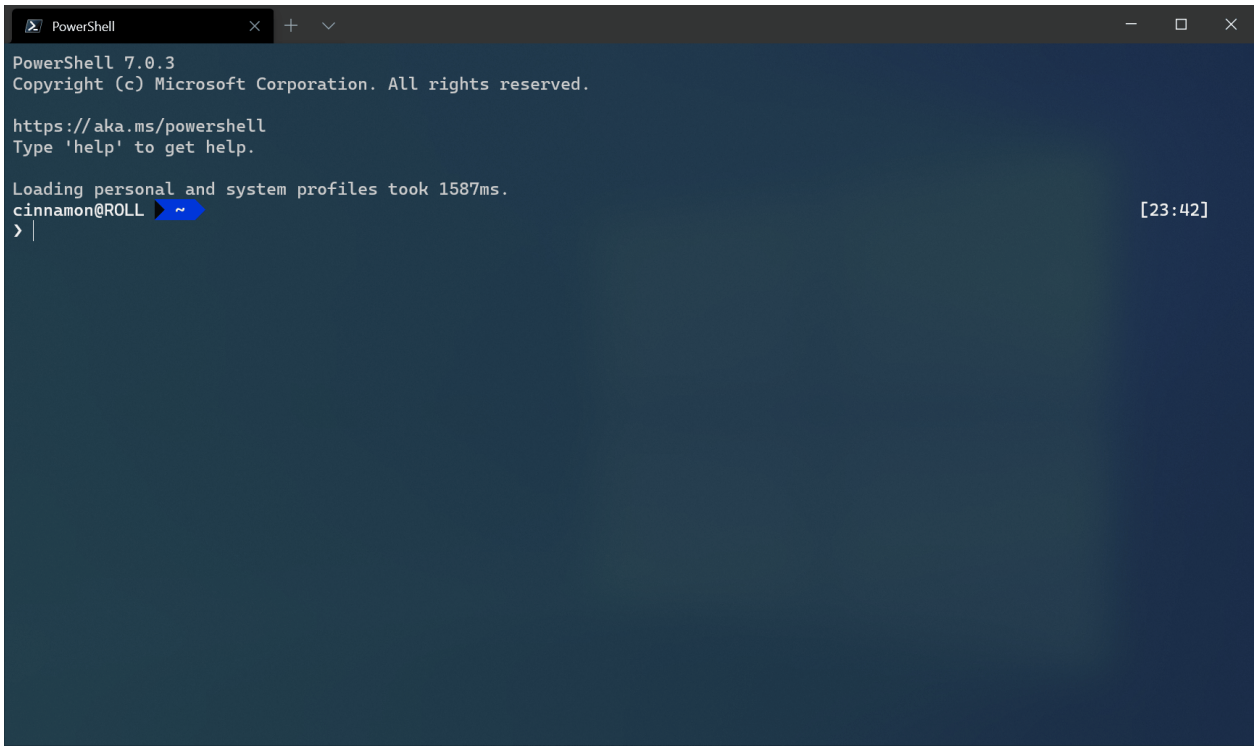
```
{
    "icon": null,
    "name": "Command Prompt",
    "command": { "action": "newTab", "profile": "Command Prompt" }
},
{
```

```
    "icon": "C:\\path\\to\\icon",
    "name": "PowerShell",
    "command": { "action": "newTab", "profile": "PowerShell" }
},
{
    "icon": null,
    "name": "Ubuntu",
    "command": { "action": "newTab", "profile": "Ubuntu" }
}
```

Il est également possible de combiner des commandes imbriquées et itérables. Par exemple, vous pouvez combiner les trois commandes "Nouvel onglet" ci-dessus sous une seule entrée "Nouvel onglet" dans la palette de commandes, comme indiqué dans l'image ci-dessus, de la manière suivante :

JSON

```
{
    "name": "New tab",
    "commands": [
        {
            "iterateOn": "profiles",
            "icon": "${profile.icon}",
            "name": "${profile.name}",
            "command": { "action": "newTab", "profile": "${profile.name}" }
        }
    ]
}
```



Masquage d’une commande

Si vous souhaitez conserver une commande dans votre liste de combinaisons de touches sans toutefois la faire apparaître dans la palette de commandes, vous pouvez la masquer en définissant `name` avec la valeur `null`. L’exemple ci-dessous masque l’action « New tab » de la palette de commandes.

JSON

```
{ "name": null, "command": "newTab", "keys": "ctrl+shift+t" }
```

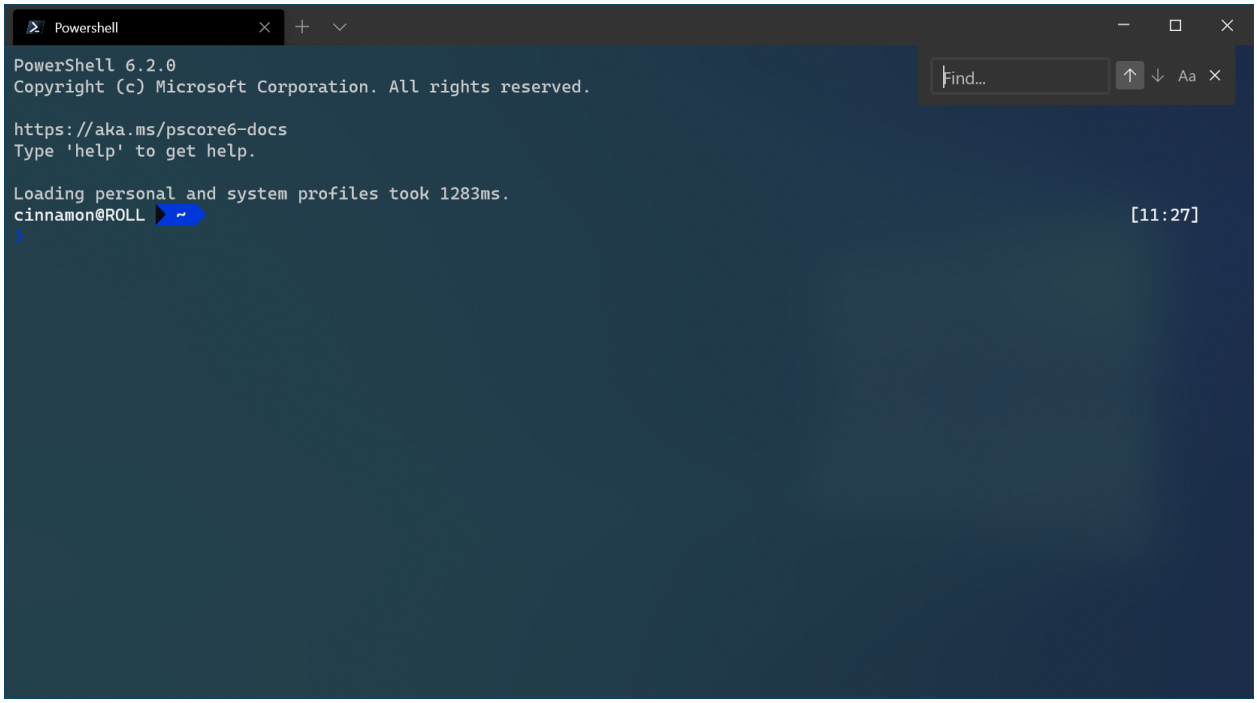
Comment effectuer une recherche dans le Terminal Windows

Article • 08/02/2024

Terminal Windows est fourni avec une fonctionnalité de recherche qui vous permet de parcourir le tampon de texte pour obtenir un mot clé spécifique. Cela est utile lorsque vous tentez de trouver une commande que vous avez exécutée avant ou pour un nom de fichier spécifique.

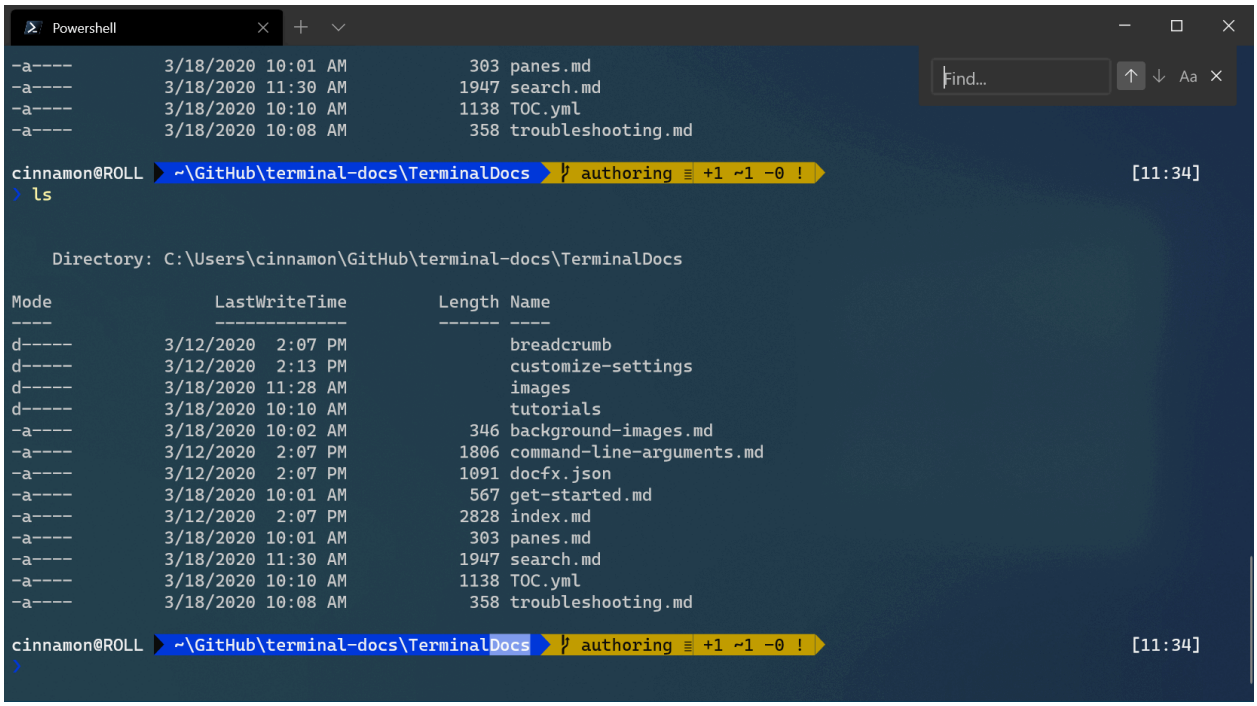
Utilisation de la recherche

Par défaut, vous pouvez ouvrir la boîte de dialogue de recherche en appuyant sur `Ctrl + Maj + F`. Une fois que vous l'avez ouverte, vous pouvez saisir le mot clé que vous recherchez dans la zone de texte, puis cliquer sur `Entrer` pour rechercher.



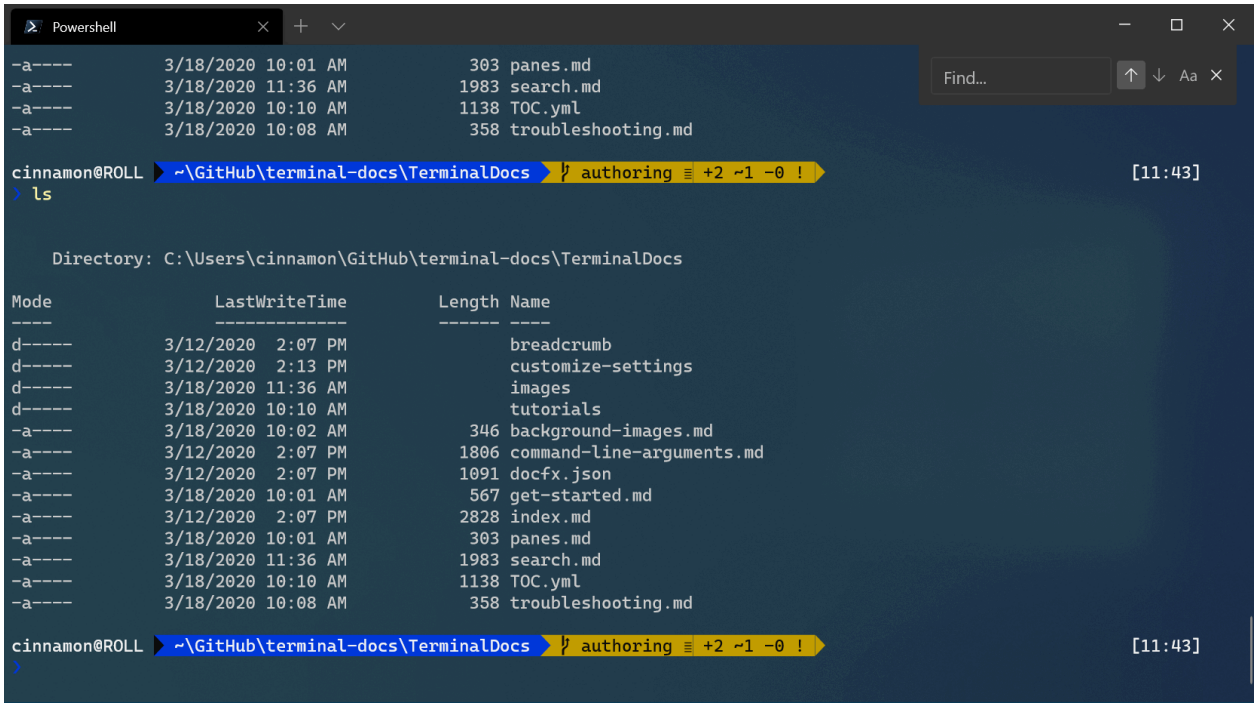
Recherche directionnelle

Par défaut, le terminal effectue du bas vers le haut du tampon de texte. Vous pouvez modifier le sens de la recherche en sélectionnant l'une des flèches dans la boîte de dialogue de recherche.



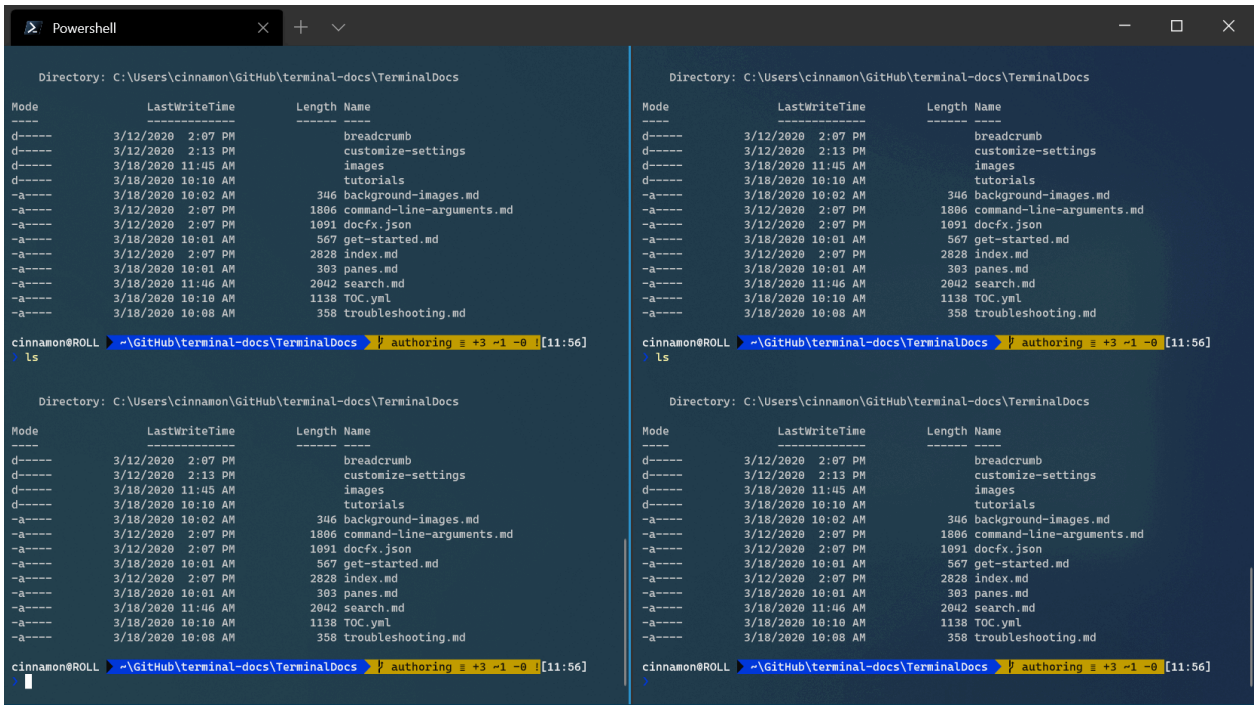
Recherche avec correspondance de casse

Si vous souhaitez affiner les résultats de votre recherche, vous pouvez ajouter la correspondance de casse en tant qu'option dans votre recherche. Vous pouvez activer ou désactiver la correspondance de la casse en sélectionnant le bouton de correspondance de casse. Les résultats qui s'affichent correspondent uniquement au mot clé entré avec cette casse spécifique.



Recherche dans les volets

La boîte de dialogue Rechercher fonctionne également avec les volets. Quand le focus est sur un volet, vous pouvez ouvrir la boîte de dialogue de recherche et celle-ci s'affichera dans le coin supérieur droit de ce volet. Le mot clé que vous saisissez affiche alors uniquement les résultats trouvés dans ce volet.



Personnaliser la combinaison de touches de recherche

Vous pouvez ouvrir la boîte de dialogue de recherche avec une combinaison de touches (un raccourci) de votre choix. Pour modifier la combinaison de touches associée à la recherche, ouvrez votre fichier settings.json et recherchez la commande `find`. Par défaut, cette commande est définie sur `Ctrl+Maj+F`.

JSON

```
// Press ctrl+shift+f to open the search box
        { "command": "find", "keys": "ctrl+shift+f" },
```

Par exemple, vous pouvez passer `"ctrl+shift+f"` à `"ctrl+f"`, par conséquent, lors de la saisie de `Ctrl+F`.

Pour en savoir plus sur les combinaisons de touches, consultez la page Actions.

Collaborer avec nous sur GitHub

La source de ce contenu se trouve sur GitHub, où vous pouvez également créer et examiner les problèmes et les demandes de tirage. Pour plus d'informations, consultez notre guide du contributeur.

Commentaires sur Windows Terminal

Windows Terminal est un projet open source. Sélectionnez un lien pour fournir des commentaires :

Ouvrir un problème de documentation

Indiquer des commentaires sur le produit

Sélection de texte dans le Terminal Windows

Article • 08/02/2024

La sélection de texte est une opération simple dans le Terminal Windows, mais il existe de nombreuses autres fonctionnalités dans cet espace qui rendent cette opération plus puissante.

Prise en charge de la souris

Cliquez avec le bouton gauche et faites glisser la souris pour créer une sélection. Un double-clic étend la sélection mot par mot, alors qu'un triple-clic l'étend ligne par ligne.

Si vous maintenez la touche `Alt` enfoncée, vous créez une sélection de bloc (par opposition à une sélection de ligne). Les sélections de bloc créent une zone rectangulaire qui n'inclut pas la fin de la ligne.

Si vous maintenez la touche `Maj` enfoncée, vous pouvez étendre explicitement la sélection jusqu'à un point spécifique dans le terminal sans avoir à cliquer-faire glisser.

Une fois que vous avez fait une sélection, vous avez plusieurs options. Un clic gauche simple efface la sélection. Si vous souhaitez réellement utiliser le texte sélectionné, vous pouvez cliquer avec le bouton droit pour copier le texte dans le Presse-papiers et ensuite effacer la sélection. Si vous recliquez avec le bouton droit, le contenu de votre Presse-papiers est collé dans le terminal.

> **Notes**
>
> Le Terminal Windows prend en charge les entrées à la souris dans les applications Sous-système Windows pour Linux (WSL) ainsi que les applications Windows qui utilisent des entrées par terminal virtuel (VT). Cela signifie que des applications comme **tmux** et **Midnight Commander** détectent quand vous sélectionnez des éléments dans la fenêtre du terminal. Si une application est en mode souris, vous pouvez maintenir la touche `Maj` enfoncée pour faire une sélection, au lieu d'envoyer une entrée VT.

Prise en charge du clavier

Vous pouvez créer une sélection à l'aide des actions `selectAll` ou `markMode`. L'action `selectAll` sélectionne tout le texte dans la mémoire tampon. L'action `markMode` active ou désactive un mode spécial où une sélection est créée à la position du curseur dans le terminal. En mode marquage, vous pouvez utiliser les combinaisons de touches non configurables suivantes pour déplacer le curseur :

Agrandir le tableau

Combinaison de touches	Résultat
Touches de direction	Déplacer caractère par caractère dans la direction spécifiée
`Ctrl+Gauche`	Déplacer au début du mot précédent ou actuel
`Ctrl+Droite`	Déplacer à la fin du mot suivant ou actuel
`Accueil`	Déplacer au début de la ligne
`End`	Déplacer à la fin de la ligne
`Pg préc`	Déplacer à la page précédente (fenêtre d'affichage)
`Pg suiv`	Déplacer à la page suivante (fenêtre d'affichage)
`Ctrl+Origine`	Déplacer au début de la mémoire tampon
`Ctrl+Fin`	Déplacer à la fin de la mémoire tampon
`Entrée`	Copier la sélection active

En mode Marquage, vous pouvez utiliser les touches `Tabulation` ou `Maj + Tabulation` pour accéder au lien hypertexte suivant ou précédent dans la mémoire tampon. Terminal Windows peut détecter automatiquement les liens hypertexte si experimental.detectUrls est activé.

Que le mode marquage soit activé ou désactivé, vous pouvez étendre une sélection existante en utilisant les combinaisons de touches non configurables suivantes :

Agrandir le tableau

Combinaison de touches	Résultat
`Maj` + touches de direction	Étendre la sélection caractère par caractère dans la direction spécifiée
`Ctrl+Maj+Gauche`	Étendre la sélection jusqu'au début du mot précédent ou actuel
`Ctrl+Maj+Droite`	Étendre la sélection jusqu'à la fin du mot suivant ou actuel

Combinaison de touches	Résultat
`Maj+Origine`	Étendre la sélection jusqu'au début de la ligne
`Maj+Fin`	Étendre la sélection jusqu'à la fin de la ligne
`Maj+Pgup`	Étendre la sélection jusqu'à la page précédente (fenêtre d'affichage)
`Maj+Pgdn`	Étendre la sélection jusqu'à la page suivante (fenêtre d'affichage)
`Ctrl+Maj+Origine`	Étendre la sélection jusqu'au début de la mémoire tampon
`Ctrl+Maj+Fin`	Étendre la sélection jusqu'à la fin de la mémoire tampon

Utilisez l'action `toggleBlockSelection` pour transformer la sélection existante en une sélection de bloc.

Toute sélection créée ou modifiée avec le clavier affiche également des marqueurs de sélection pour indiquer quelle extrémité de la sélection fait l'objet du déplacement. Vous pouvez utiliser l'action `switchSelectionEndpoint` pour commencer à déplacer la sélection vers l'autre extrémité de la sélection.

Une fois que vous avez fait une sélection, vous avez plusieurs options. Vous pouvez utiliser la touche `Échap` pour effacer la sélection. Sinon, la plupart des touches effacent la sélection et passent l'événement touche directement au shell sous-jacent. Si vous souhaitez réellement utiliser le texte sélectionné, vous pouvez utiliser l'action `copy` pour copier le texte dans le Presse-papiers.

Copie du texte sélectionné

Comme mentionné ci-dessus, vous pouvez copier le texte sélectionné avec un clic droit ou l'action `copy`. Toutefois, il existe un certain nombre de paramètres de copie de texte que vous pouvez personnaliser :

- Copie du texte mis en forme
  - Vous pouvez utiliser le paramètre général `copyFormatting` pour copier à la fois le texte sélectionné et sa mise en forme dans le Presse-papiers. Cela vous permet de copier les informations de police du terminal, telles que la couleur de premier plan, la couleur d'arrière-plan et la police.
  - Si vous souhaitez limiter la copie de la mise en forme à certaines combinaisons de touches (ou commandes), vous pouvez modifier le paramètre `copyFormatting` sur une action `copy`.
- Copier sans ignorer la sélection de texte

    - Vous pouvez copier du texte sans ignorer la sélection de texte en définissant le paramètre `dismissSelection` de l'action `copy` sur `false`.
- Copie sous forme d'une seule ligne
    - Vous pouvez copier du texte sous la forme d'une seule ligne en utilisant le paramètre `singleLine` dans l'action `copy`.
- Suppression de l'espace blanc de fin dans les sélections de bloc
    - Vous pouvez supprimer l'espace blanc de fin dans une sélection de bloc en utilisant le paramètre général `trimBlockSelection`.

Vous pouvez également utiliser le paramètre général `copyOnSelect` pour que tout nouveau texte sélectionné soit automatiquement copié dans votre Presse-papiers. Lorsque ce paramètre est activé, si une sélection existe déjà et que vous cliquez avec le bouton droit dans le terminal, le texte sélectionné est copié et collé dans le terminal.

> ⓘ **Notes**
>
> Si `copyOnSelect` est activé, la modification de la sélection avec le clavier n'entraîne pas automatiquement la copie du nouveau texte sélectionné. Vous devez copier manuellement le texte en utilisant l'action `copy` ou en cliquant avec le bouton droit dans le terminal.

Personnalisation de l'apparence des sélections

Dans les modèles de couleurs, vous pouvez personnaliser la couleur de sélection en utilisant la propriété `selectionBackground`. Vous pouvez également remplacer la couleur de sélection d'un profil spécifique en modifiant le paramètre `selectionBackground` du profil.

Personnalisation des délimiteurs de mots

Comme mentionné plus haut, vous pouvez utiliser le double-clic et les combinaisons de touches `Ctrl + Maj` + touches de direction (ou `CtrlMaj` + touches de direction en mode marquage) pour vous déplacer d'un mot à un autre. Toutefois, les mots peuvent être séparés par plusieurs espaces blancs. Vous pouvez personnaliser ces délimiteurs de mots en utilisant le paramètre général `wordDelimiters`.

Collaborer avec nous sur GitHub

La source de ce contenu se trouve sur GitHub, où vous pouvez également créer et examiner les problèmes et les demandes de tirage. Pour plus d'informations, consultez notre guide du contributeur.

Commentaires sur Windows Terminal

Windows Terminal est un projet open source. Sélectionnez un lien pour fournir des commentaires :

Ouvrir un problème de documentation

Indiquer des commentaires sur le produit

Volets dans le Terminal Windows

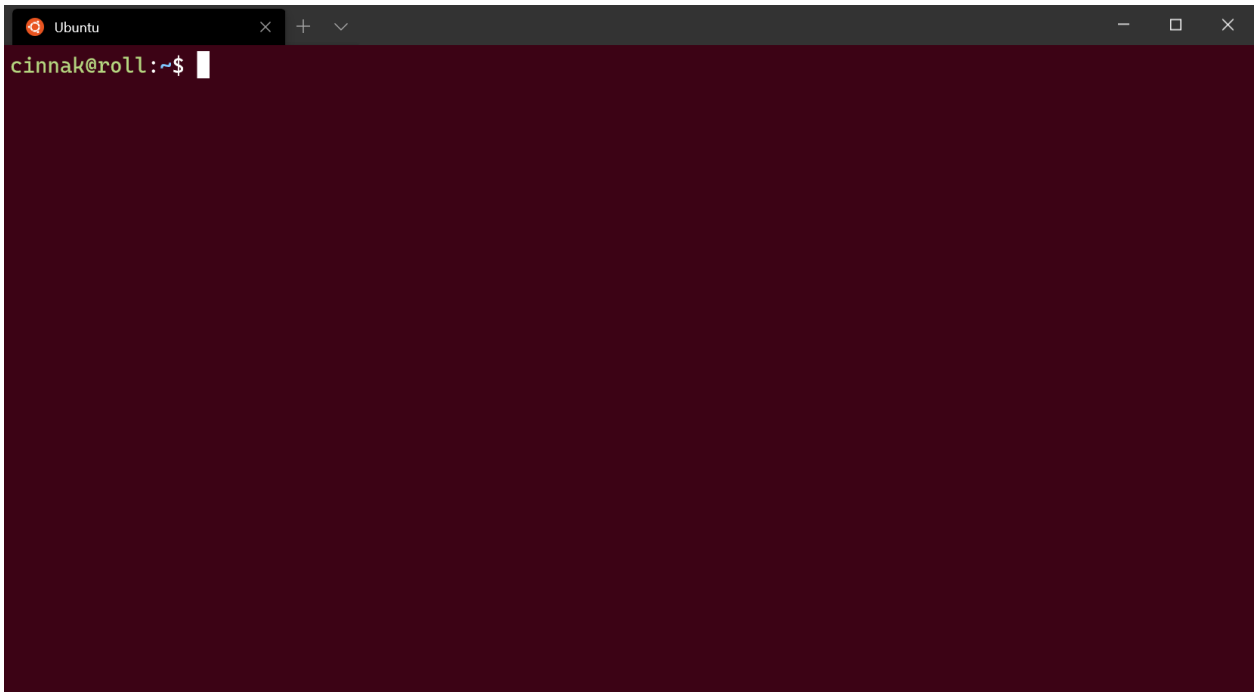
Article • 04/10/2023

Les volets vous permettent d'exécuter plusieurs applications en ligne de commande les unes à côté des autres dans le même onglet. Cela réduit le besoin de basculer entre les onglets et vous permet d'afficher plusieurs invites à la fois.

Création d'un nouveau volet

Utilisation du clavier

Vous pouvez créer un volet vertical ou horizontal dans Terminal Windows. Le fractionnement vertical ouvre un nouveau volet à droite du volet ayant le focus, et le fractionnement horizontal ouvre un nouveau volet sous le volet ayant le focus. Les divisions directionnelles `up`, `right`, `down` et `left` vous donnent d'autres options de placement pour votre nouveau volet. `right` et `down` sont équivalents à `vertical` et `horizontal`, tandis que `up` et `left` vous permettent de placer le nouveau volet respectivement au-dessus et à gauche du volet qui a le focus. Pour créer un volet vertical de votre profil par défaut, vous pouvez utiliser la combinaison de touches `Alt`+`Maj`+`+`. Pour créer un volet horizontal de votre profil par défaut, utilisez `Alt`+`Maj`+`-`.



Configuration : Raspberry Ubuntu

Si vous souhaitez modifier ces combinaisons de touches, vous pouvez en créer d'autres en utilisant l'action `splitPane` et les valeurs `vertical`, `horizontal`, `up`, `right`, `down`,

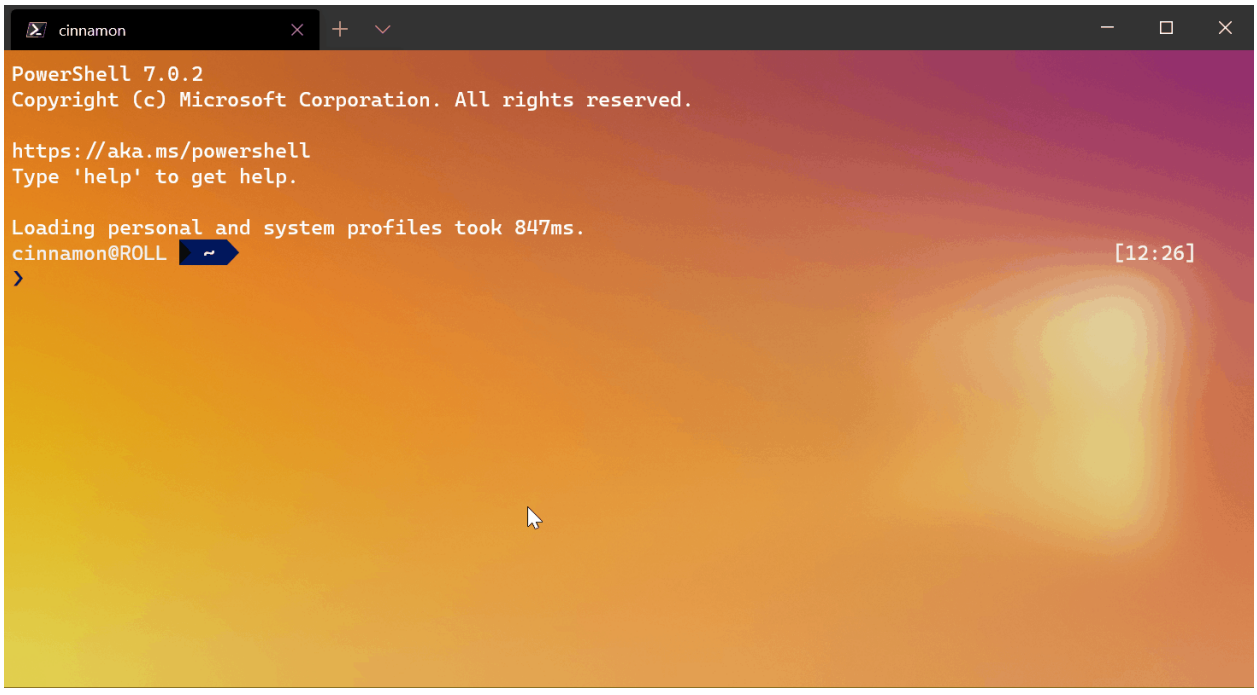
`left` ou `auto` pour la propriété `split` dans votre fichier profiles.json. La méthode `auto` choisit la direction qui vous donne les volets les plus carrés. Pour en savoir plus sur les combinaisons de touches, consultez la page Actions.

JSON

```
{ "command": { "action": "splitPane", "split": "vertical" }, "keys":
"alt+shift+plus" },
{ "command": { "action": "splitPane", "split": "horizontal" }, "keys":
"alt+shift+-" },
{ "command": { "action": "splitPane", "split": "auto" }, "keys":
"alt+shift+d" },
{ "command": { "action": "splitPane", "split": "up" } },
{ "command": { "action": "splitPane", "split": "right" } },
{ "command": { "action": "splitPane", "split": "down" } },
{ "command": { "action": "splitPane", "split": "left" } },
```

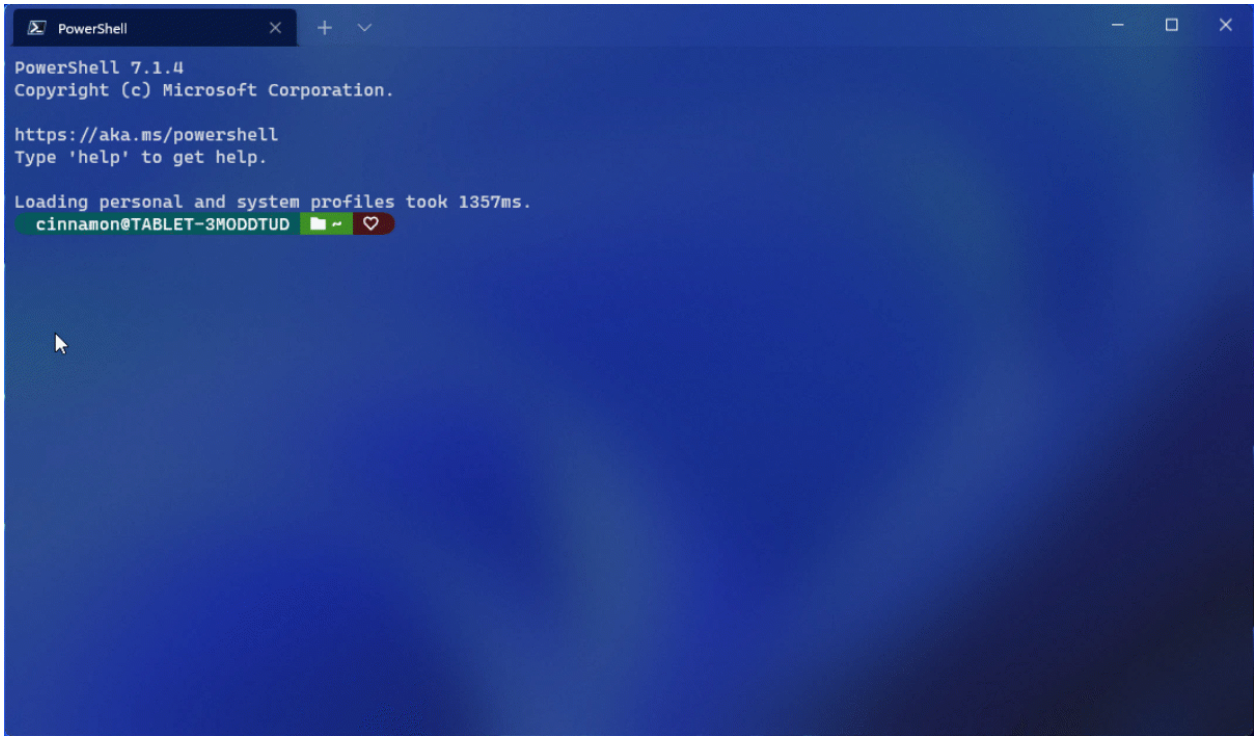
Utilisation du bouton Nouvel onglet et du menu déroulant

Si vous souhaitez ouvrir un nouveau volet de votre profil par défaut, vous pouvez maintenir la touche `Alt` enfoncée et cliquer sur le bouton Nouvel onglet. Si vous souhaitez ouvrir un nouveau volet via le menu déroulant, vous pouvez maintenir la touche `Alt` enfoncée et cliquer sur le profil de votre choix. Ces deux options divisent en mode `auto` la fenêtre ou le volet actif dans un nouveau volet du profil sélectionné. Le mode fractionné `auto` effectue le fractionnement dans la direction avec le bord le plus long pour créer un volet.



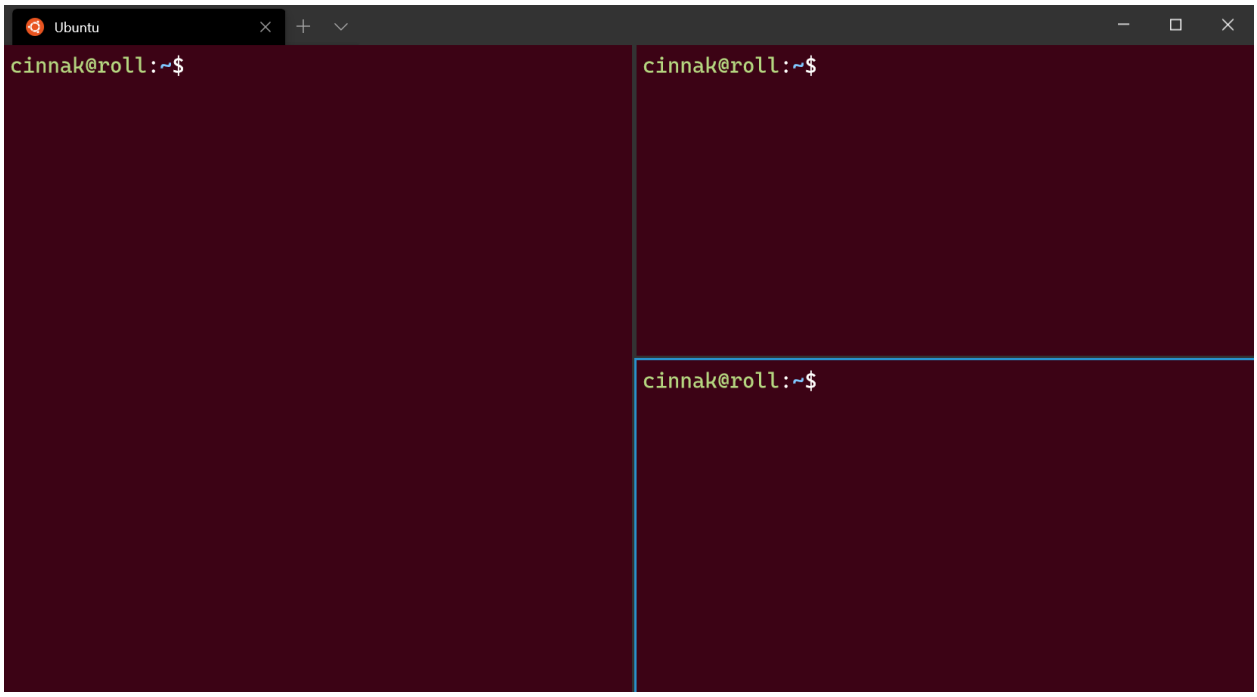
Utilisation du menu contextuel de l’onglet

Si vous souhaitez ouvrir un nouveau volet d’un profil déjà ouvert dans votre terminal, vous pouvez cliquer avec le bouton droit sur l’onglet et cliquer sur Diviser l’onglet. Le volet ayant le focus est alors dupliqué dans l’onglet actif.



Basculement entre les volets

Le terminal vous permet de naviguer entre les volets à l’aide du clavier. Si vous maintenez la touche `Alt` enfoncée, vous pouvez utiliser les touches de direction pour déplacer le focus entre les volets. Vous pouvez identifier le volet qui est activé par la bordure de couleur d’accentuation qui l’entoure. Notez que cette couleur d’accentuation est définie dans les paramètres de couleurs de Windows.



Vous pouvez personnaliser ce comportement en ajoutant des combinaisons de touches pour la commande `moveFocus` et en définissant `direction` sur `down`, `left`, `right` ou `up`. De plus, `direction` peut avoir la valeur `previous` pour le dernier volet utilisé, `previousInOrder` ou `nextInOrder` pour la navigation par ordre dans l'arborescence, ou `first` pour le premier volet. Enfin, vous pouvez monter dans l'arborescence en utilisant la direction `parent` pour sélectionner plusieurs volets, puis utiliser `child` pour sélectionner moins de volets et redescendre dans l'arborescence. Quand plusieurs volets sont sélectionnés, vous pouvez effectuer des actions comme si un seul volet avait le focus.

JSON

```
{ "command": { "action": "moveFocus", "direction": "down" }, "keys":
"alt+down" },
{ "command": { "action": "moveFocus", "direction": "left" }, "keys":
"alt+left" },
{ "command": { "action": "moveFocus", "direction": "right" }, "keys":
"alt+right" },
{ "command": { "action": "moveFocus", "direction": "up" }, "keys": "alt+up"
},
{ "command": { "action": "moveFocus", "direction": "previous" } },
{ "command": { "action": "moveFocus", "direction": "previousInOrder" } },
{ "command": { "action": "moveFocus", "direction": "nextInOrder" } },
{ "command": { "action": "moveFocus", "direction": "first" } },
{ "command": { "action": "moveFocus", "direction": "parent" } },
{ "command": { "action": "moveFocus", "direction": "child" } }
```

Permutation de volets

Après avoir créé deux volets, vous pouvez permuter leurs positions dans le terminal.

La commande `swapPane` peut être personnalisée avec les mêmes options de `direction` que `moveFocus`, à l'exception de `parent` et de `child`. Ces commandes permettent de permuter les positions du volet ayant le focus et de son voisin en fonction de la `direction`.

JSON

```
{ "command": { "action": "swapPane", "direction": "down" } },
{ "command": { "action": "swapPane", "direction": "left" } },
{ "command": { "action": "swapPane", "direction": "right" } },
{ "command": { "action": "swapPane", "direction": "up" } },
{ "command": { "action": "swapPane", "direction": "previous" } },
{ "command": { "action": "swapPane", "direction": "previousInOrder" } },
{ "command": { "action": "swapPane", "direction": "nextInOrder" } },
{ "command": { "action": "swapPane", "direction": "first" } }
```

Déplacement de volets

Vous pouvez également déplacer des volets d'un onglet à l'autre. Si un onglet avec l'index cible n'existe pas, il est créé.

Les combinaisons de touches pour la commande `movePane` peuvent être personnalisées pour déplacer des volets vers des onglets (indexés sur zéro) en fonction de leur ordre.

JSON

```
{ "command": { "action": "movePane", "index": 0 } },
{ "command": { "action": "movePane", "index": 1 } },
{ "command": { "action": "movePane", "index": 2 } },
{ "command": { "action": "movePane", "index": 3 } },
{ "command": { "action": "movePane", "index": 4 } },
{ "command": { "action": "movePane", "index": 5 } },
{ "command": { "action": "movePane", "index": 6 } },
{ "command": { "action": "movePane", "index": 7 } },
{ "command": { "action": "movePane", "index": 8 } }
```

Modification de l'orientation de la division

Une fois un onglet avec deux volets créé, vous pouvez changer l'orientation de la division appliquée à ces volets (`vertical` ou `horizontal`) avec la commande `toggleSplitOrientation`.

JSON

```
{ "command": "toggleSplitOrientation" }
```

Permutation de volets (préversion)

Après avoir créé deux volets, vous pouvez permuter leurs positions dans le terminal.

La commande `swapPane` peut être personnalisée avec les mêmes options de `direction` que `moveFocus`. Ces commandes permettent de permuter les positions du volet ayant le focus et de son voisin en fonction de la `direction`.

JSON

```
{ "command": { "action": "swapPane", "direction": "down" } },
{ "command": { "action": "swapPane", "direction": "left" } },
{ "command": { "action": "swapPane", "direction": "right" } },
{ "command": { "action": "swapPane", "direction": "up" } },
{ "command": { "action": "swapPane", "direction": "previous" } },
{ "command": { "action": "swapPane", "direction": "previousInOrder" } },
{ "command": { "action": "swapPane", "direction": "nextInOrder" } }
```

Déplacement de volets (préversion)

Vous pouvez également déplacer des volets d'un onglet à l'autre. Si un onglet avec l'index cible n'existe pas, il est créé.

Les combinaisons de touches pour la commande `movePane` peuvent être personnalisées pour déplacer des volets vers des onglets (indexés sur zéro) en fonction de leur ordre.

JSON

```
{ "command": { "action": "movePane", "index": 0 } },
{ "command": { "action": "movePane", "index": 1 } },
{ "command": { "action": "movePane", "index": 2 } },
{ "command": { "action": "movePane", "index": 3 } },
{ "command": { "action": "movePane", "index": 4 } },
{ "command": { "action": "movePane", "index": 5 } },
{ "command": { "action": "movePane", "index": 6 } },
{ "command": { "action": "movePane", "index": 7 } },
{ "command": { "action": "movePane", "index": 8 } }
```

Modification de l'orientation de la division (préversion)

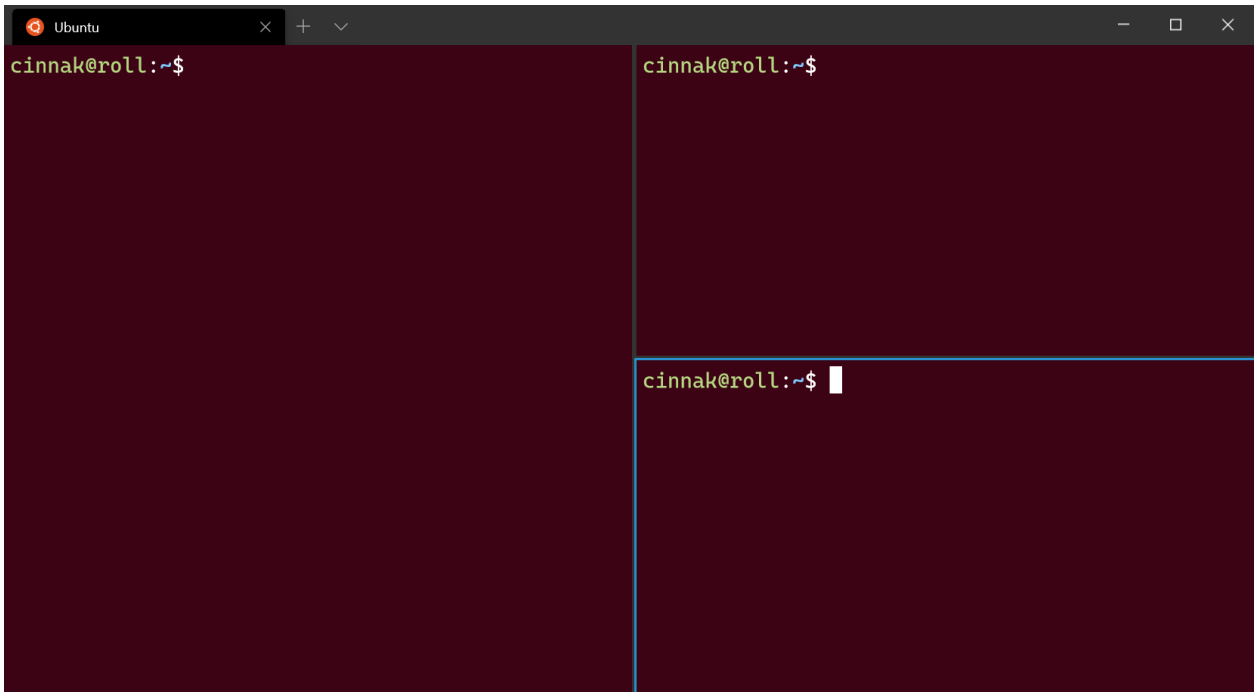
Une fois un onglet avec deux volets créé, vous pouvez changer l'orientation de la division appliquée à ces volets (`vertical` ou `horizontal`) avec la commande `toggleSplitOrientation`.

JSON

```
{ "command": "toggleSplitOrientation" }
```

Redimensionnement d'un volet

Vous pouvez ajuster la taille de vos volets en maintenant les touches `Alt`+`Maj` enfoncées et en utilisant les touches de direction pour redimensionner le volet qui a le focus.



Pour personnaliser cette combinaison de touches, vous pouvez en ajouter de nouvelles à l'aide de l'action `resizePane` et définir `direction` sur `down`, `left`, `right` ou `up`.

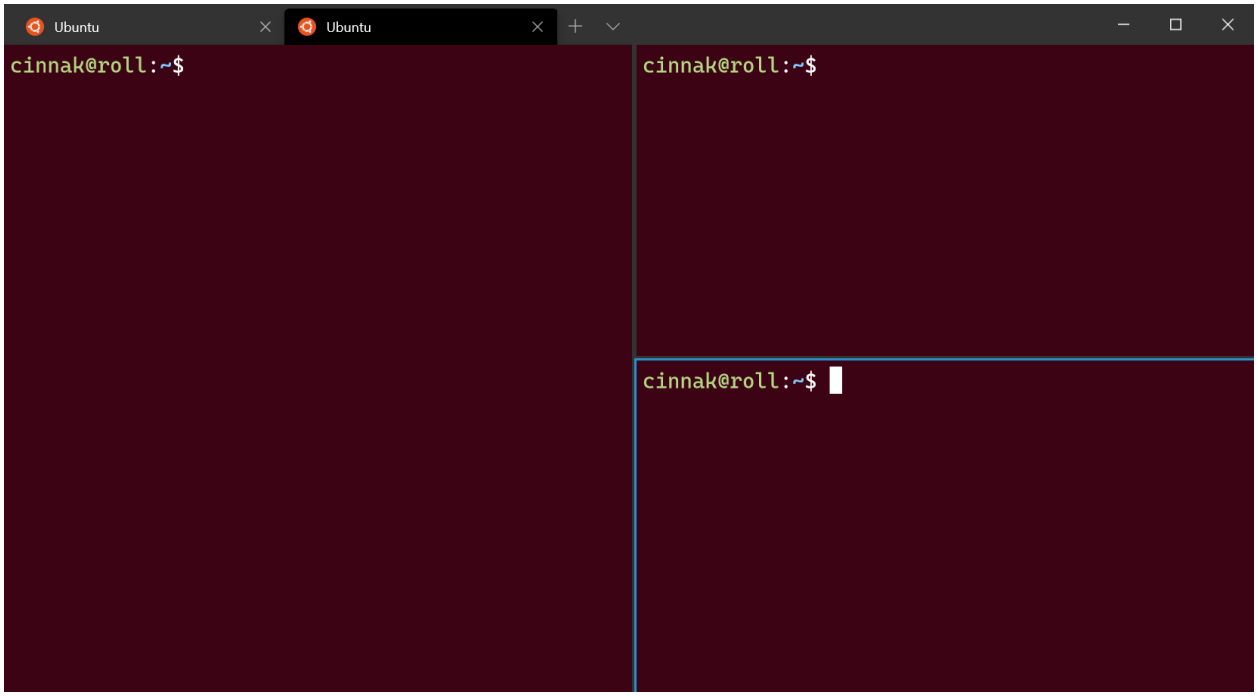
JSON

```
{ "command": { "action": "resizePane", "direction": "down" }, "keys":
"alt+shift+down" },
{ "command": { "action": "resizePane", "direction": "left" }, "keys":
"alt+shift+left" },
{ "command": { "action": "resizePane", "direction": "right" }, "keys":
```

```
"alt+shift+right" },
{ "command": { "action": "resizePane", "direction": "up" }, "keys":
"alt+shift+up" }
```

Fermeture d'un volet

Vous pouvez fermer le volet qui a le focus en appuyant sur `Ctrl`+`Maj`+`W`. Si vous n'avez qu'un seul volet, la combinaison de touches `Ctrl`+`Maj`+`W` ferme l'onglet. Comme toujours, la fermeture du dernier onglet entraîne la fermeture de la fenêtre.



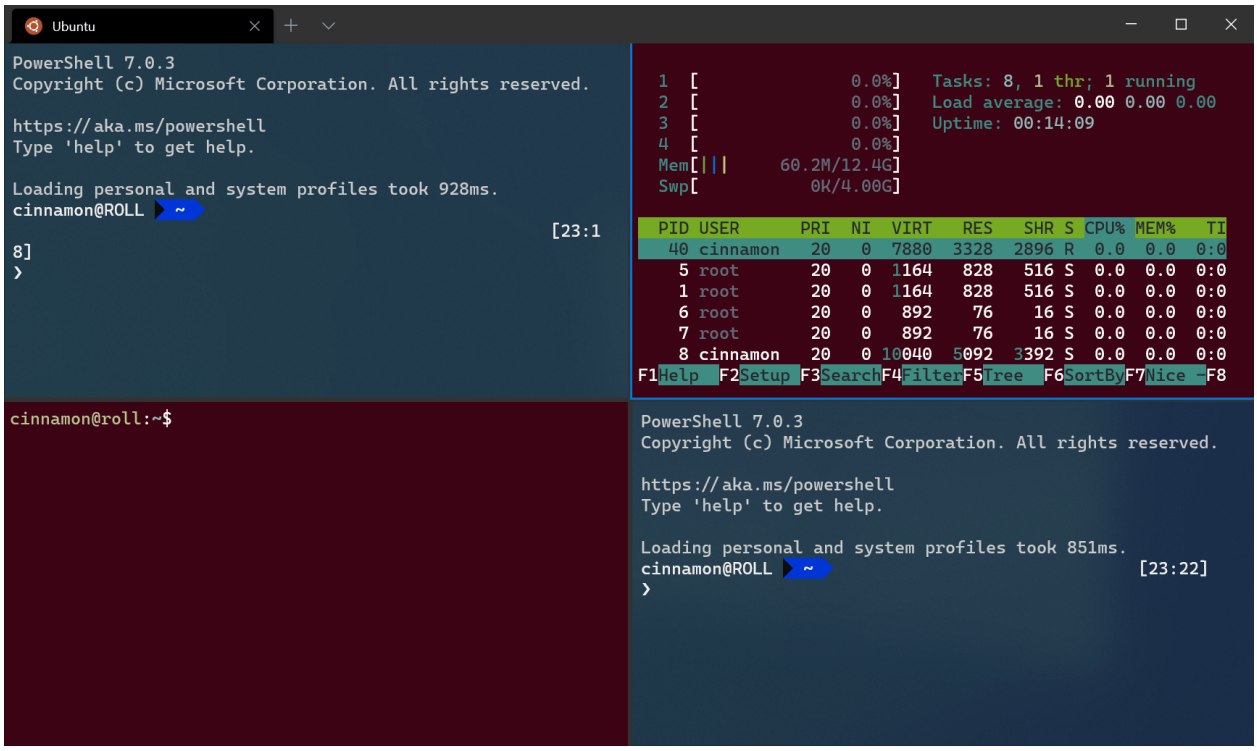
Vous pouvez modifier les touches qui ferment le volet en ajoutant une combinaison de touches qui utilise la commande `closePane`.

JSON

```
{ "command": "closePane", "keys": "ctrl+shift+w" }
```

Zoom sur un volet

Vous pouvez faire un zoom sur le volet ayant le focus pour remplir tout le contenu de la fenêtre.



> ⓘ Notes
>
> Le menu dans l'illustration ci-dessus est la **Palette de commandes**, qui peut être ouverte par défaut avec `Ctrl+Maj+P`.

Cette opération peut être effectuée au moyen de la commande `togglePaneZoom`.

JSON

```
{ "command": "togglePaneZoom" }
```

> ⓘ Notes
>
> L'action `togglePaneZoom` n'est liée à aucune combinaison de touches par défaut, mais elle est accessible par le biais de la **palette de commandes** liée à `Ctrl`+`Maj`+`P` par défaut.

Marquage d'un volet en lecture seule

Vous pouvez marquer un volet comme étant en lecture seule, ce qui a pour effet d'empêcher le transfert de toute entrée dans la mémoire tampon de texte. Si vous tentez de fermer ou d'entrer du texte dans un volet en lecture seule, le terminal affiche un avertissement contextuel.

Vous pouvez activer ou désactiver le mode lecture seule sur un volet avec la commande `toggleReadOnlyMode`.

```JSON
{ "command": "toggleReadOnlyMode" }
```

Vous pouvez activer le mode lecture seule dans un volet. Cela fonctionne de la même façon qu'un bouton bascule, mais par contre, cela ne change pas l'état si redéclenché.

Nom de la commande : `enableReadOnlyMode`

Copier les liaisons par défaut :

```JSON
{ "command": "enableReadOnlyMode" }
```

Vous pouvez désactiver le mode lecture seule dans un volet. Cela fonctionne de la même façon qu'un bouton bascule, mais par contre, cela ne change pas l'état si redéclenché.

Nom de la commande : `disableReadOnlyMode`

Copier les liaisons par défaut :

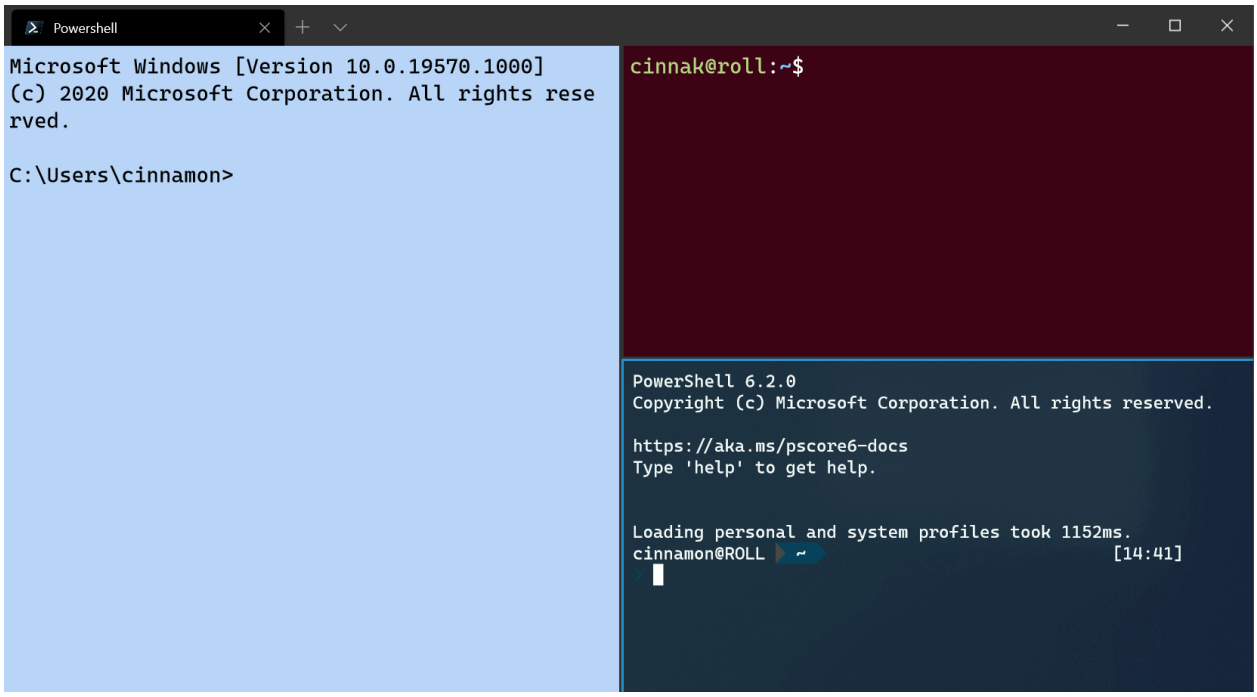
```JSON
{ "command": "disableReadOnlyMode" }
```

Personnalisation des volets à l'aide de combinaisons de touches

Vous pouvez personnaliser ce qui s'ouvre dans un nouveau volet en fonction de vos combinaisons de touches personnalisées.

Duplication d'un volet

Le terminal vous permet de dupliquer le profil du volet qui a le focus dans un autre volet.



Pour ce faire, vous pouvez ajouter la propriété `splitMode` avec `duplicate` comme valeur à une combinaison de touches `splitPane`.

JSON

```
{ "command": { "action": "splitPane", "split": "auto", "splitMode":
"duplicate" }, "keys": "alt+shift+d" }
```

Nouveaux arguments de terminal

Lors de l'ouverture d'un nouveau volet ou d'un nouvel onglet avec une combinaison de touches, vous pouvez spécifier le profil utilisé en incluant le nom du profil, le GUID ou l'index. Si vous n'en spécifiez aucun, le profil par défaut est utilisé. Pour ce faire, vous pouvez ajouter `profile` ou `index` en tant qu'argument à une combinaison de touches `splitPane` ou `newTab`. Notez que l'indexation commence à 0.

JSON

```
{ "command": { "action": "splitPane", "split": "vertical", "profile":
"profile1" }, "keys": "ctrl+a" },
{ "command": { "action": "splitPane", "split": "vertical", "profile": "
{00000000-0000-0000-0000-000000000000}" }, "keys": "ctrl+b" },
{ "command": { "action": "newTab", "index": 0 }, "keys": "ctrl+c" }
```

En outre, vous pouvez remplacer certains aspects du profil, tels que l'exécutable de ligne de commande du profil, le répertoire de départ ou le titre de l'onglet. Pour ce faire, vous

pouvez ajouter `commandline`, `startingDirectory` et/ou `tabTitle` à une combinaison de touches de `splitPane` ou `newTab`.

JSON

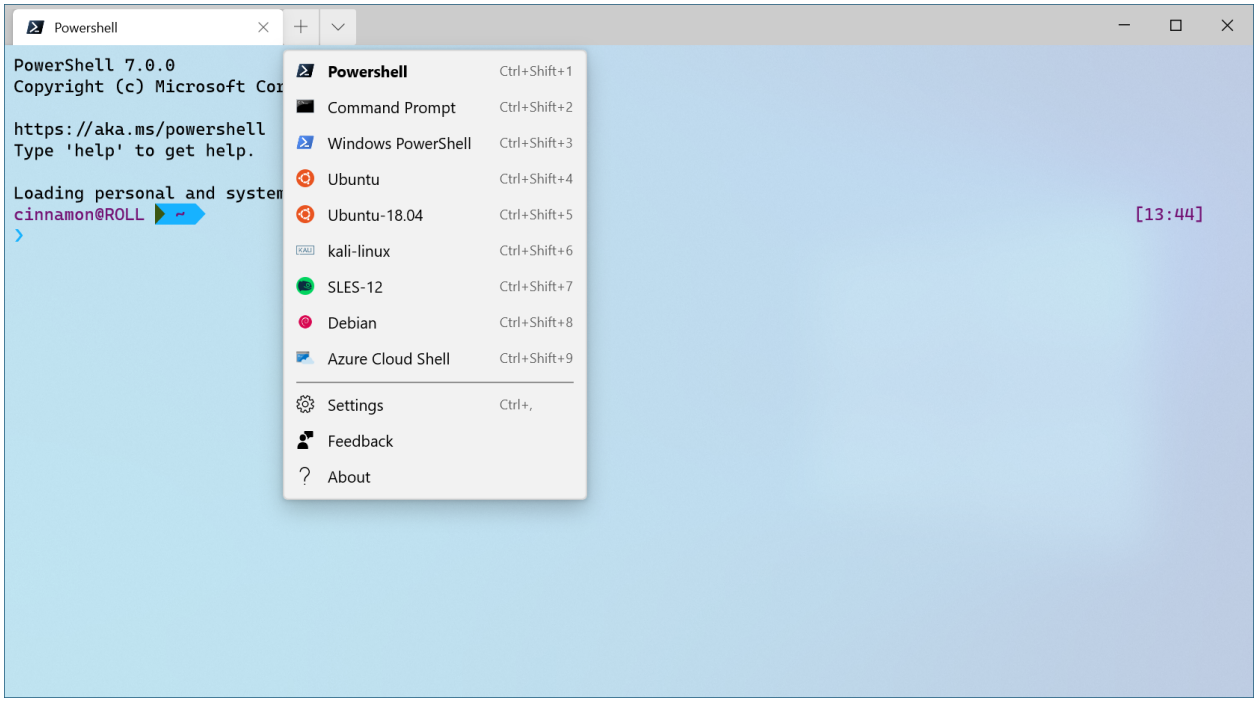
```
{ "command": { "action": "splitPane", "split": "auto", "profile":
"profile1", "commandline": "foo.exe" }, "keys": "ctrl+a" },
{ "command": { "action": "newTab", "profile": "{00000000-0000-0000-0000-
000000000000}", "startingDirectory": "C:\\foo" }, "keys": "ctrl+b" },
{ "command": { "action": "newTab", "index": 0, "tabTitle": "bar",
"startingDirectory": "C:\\foo", "commandline": "foo.exe" }, "keys": "ctrl+c"
}
```

Profils dynamiques du Terminal Windows

Article • 21/03/2023

Terminal Windows crée automatiquement pour vous un sous-système Windows pour les profils Linux (WSL) et PowerShell si ces interpréteurs de commandes (shells) sont installés sur votre ordinateur. Cela facilite l'inclusion de tous vos interpréteurs de commandes dans le terminal sans avoir à localiser leurs fichiers exécutables. Ces profils sont générés avec la propriété `source`, qui indique au terminal où trouver le fichier exécutable approprié.

Lors de l'installation du terminal, il définit PowerShell comme votre profil par défaut. Pour savoir comment modifier votre profil par défaut, accédez à la page de démarrage.



Configuration : Thème clair

Installation d'un nouvel interpréteur de commandes après l'installation du Terminal Windows

Qu'un nouvel interpréteur de commandes soit installé avant ou après l'installation de votre terminal, le terminal crée un nouveau profil pour l'interpréteur de commandes qui vient d'être installé.

Masquer un profil

Pour masquer un profil dans le menu déroulant de votre terminal, ajoutez la propriété `hidden` à l'objet de profil dans votre fichier settings.json et affectez-lui la valeur `true`.

JSON

```
"hidden": true
```

Empêcher la génération d'un profil

Pour empêcher la génération d'un profil dynamique, vous pouvez ajouter le générateur de profils au tableau de `disabledProfileSources` dans vos paramètres globaux. Pour plus d'informations sur ce paramètre, consultez la page Paramètres globaux.

JSON

```
"disabledProfileSources": ["Windows.Terminal.Wsl", "Windows.Terminal.Azure",
"Windows.Terminal.PowershellCore", "Windows.Terminal.SSH"]
```

Ajouter un profil tiers

Si un outil en ligne de commande tiers n'a pas de profil généré automatiquement dans votre fichier settings.json, vous pouvez l'ajouter manuellement. Vous trouverez ci-dessous les profils de quelques outils tiers courants pour référence.

Anaconda

En supposant que vous avez installé Anaconda dans `%USERPROFILE%\Anaconda3` :

JSON

```
{
    "commandline": "cmd.exe /k
\"%USERPROFILE%\\Anaconda3\\Scripts\\activate.bat
%USERPROFILE%\\Anaconda3\"",
    "icon": "%USERPROFILE%\\Anaconda3\\Menu\\anaconda-navigator.ico",
    "name": "Anaconda3",
    "startingDirectory": "%USERPROFILE%"
}
```

cmder

En supposant que vous avez installé cmder dans `%CMDER_ROOT%` :

JSON

```
{
    "commandline": "cmd.exe /k \"%CMDER_ROOT%\\vendor\\init.bat\"",
    "name": "cmder",
    "icon": "%CMDER_ROOT%\\icons\\cmder.ico",
    "startingDirectory": "%USERPROFILE%"
}
```

Cygwin

En supposant que vous avez installé Cygwin dans `C:\Cygwin` :

JSON

```
{
    "name": "Cygwin",
    "commandline": "C:\\Cygwin\\bin\\bash --login -i",
    "icon": "C:\\Cygwin\\Cygwin.ico",
    "startingDirectory": "C:\\Cygwin\\bin"
}
```

> Remarque : Le répertoire de départ de Cygwin est défini pour que le chemin fonctionne. Le répertoire par défaut ouvert au démarrage de Cygwin sera `$HOME` en raison de l'indicateur `--login`.

Far Manager

En supposant que vous avez installé Far dans `c:\Program Files\Far Manager` :

JSON

```
{
    "name": "Far",
    "commandline": "\"c:\\program files\\far manager\\far.exe\"",
    "startingDirectory": "%USERPROFILE%",
    "useAcrylic": false
},
```

Git Bash

En supposant que vous avez installé Git Bash dans `C:\\Program Files\\Git` :

JSON

```
{
    "name": "Git Bash",
    "commandline": "C:\\Program Files\\Git\\bin\\bash.exe -li",
    "icon": "C:\\Program Files\\Git\\mingw64\\share\\git\\git-for-
windows.ico",
    "startingDirectory": "%USERPROFILE%"
}
```

Git Bash (WOW64)

En supposant que vous avez installé Git Bash dans `C:\\Program Files (x86)\\Git` :

JSON

```
{
    "name": "Git Bash",
    "commandline": "%ProgramFiles(x86)%\\Git\\bin\\bash.exe -li",
    "icon": "%ProgramFiles(x86)%\\Git\\mingw32\\share\\git\\git-for-
windows.ico",
    "startingDirectory": "%USERPROFILE%"
}
```

MSYS2

En supposant que vous avez installé MSYS2 dans `C:\\msys64` :

JSON

```
{
    "name": "MSYS2",
    "commandline": "C:\\msys64\\msys2_shell.cmd -defterm -no-start -
mingw64",
    "icon": "C:\\msys64\\msys2.ico",
    "startingDirectory": "C:\\msys64\\home\\user"
}
```

Pour plus d'informations, consultez la section Terminals de la documentation MSYS2.

Extensions de fragments JSON dans le Terminal Windows

Article • 21/03/2023

Les extensions de fragments JSON sont des extraits de code JSON que les développeurs d'applications peuvent écrire pour ajouter de nouveaux profils aux paramètres des utilisateurs et même modifier certains profils existants. Elles permettent également d'ajouter de nouveaux modèles de couleurs aux paramètres des utilisateurs.

Structure des fichiers JSON

Le fichier JSON doit être divisé en deux listes : une pour les profils et une pour les modèles. Voici un exemple de fichier JSON qui ajoute un nouveau profil, modifie un profil existant et crée un modèle de couleurs :

JSON

```
{
  "profiles": [
    {
      // update a profile by using its GUID
      "updates": "{2ece5bfe-50ed-5f3a-ab87-5cd4baafed2b}",
      "fontSize": 16,
      "fontWeight": "thin"
    },
    {
      // create a new profile
      "name": "Cool Profile",
      "commandline": "powershell.exe",
      "antialiasingMode": "aliased",
      "fontWeight": "bold",
      "colorScheme": "Postmodern Tango Light"
    }
  ],
  "schemes": [
    {
      // create a new color scheme
      "name": "Postmodern Tango Light",
      "black": "#0C0C0C",
      "red": "#C50F1F",
      "green": "#13A10E",
      "yellow": "#C19C00",
      "blue": "#0037DA",
      "purple": "#881798",
      "cyan": "#3A96DD",
      "white": "#CCCCCC",
      "brightBlack": "#767676",
```

```
      "brightRed": "#E74856",
      "brightGreen": "#16C60C",
      "brightYellow": "#F9F1A5",
      "brightBlue": "#3B78FF",
      "brightPurple": "#B4009E",
      "brightCyan": "#61D6D6",
      "brightWhite": "#F2F2F2"
    }
  ]
}
```

Le premier élément de la liste `"profiles"` met à jour un profil existant, celui-ci étant identifié par le GUID fourni au champ `"updates"`. Vous verrez comment obtenir le GUID dans la section suivante. Le deuxième élément de cette liste crée un profil appelé « Cool Profile ».

Dans la liste `"schemes"`, un nouveau modèle de couleurs appelé « Postmodern Tango Light » est défini. L'utilisateur peut par la suite référencer ce modèle dans son fichier de paramètres ou dans ce fichier JSON (notez que « Cool Profile » utilise le nouveau modèle de couleurs défini).

Bien entendu, si le développeur souhaite uniquement ajouter/modifier des profils sans ajouter de modèles de couleurs (et inversement), seule la liste appropriée doit être présente et l'autre liste peut être omise.

> ⓘ **Notes**
>
> Si vous envisagez d'utiliser PowerShell pour générer des fragments, vous devez utiliser `-Encoding Utf8` :
>
> PowerShell
>
> ```
> # BAD: PowerShell uses UTF16LE by default
> Write-Output $fragmentJson > $fragmentPath
> ```
>
> PowerShell
>
> ```
> # GOOD: Uses UTF8, so Terminal will read this
> Write-Output $fragmentJson | Out-File $fragmentPath -Encoding Utf8
> ```
>
> Si vous utilisez VS Code pour modifier du code JSON, UTF8 est la valeur par défaut (ce que vous pouvez confirmer dans la barre d'état inférieure).

GUID de profil

Comme indiqué précédemment, les GUID de profil sont un moyen de référencer les profils et de permettre aux utilisateurs de les mettre à jour et de les étendre sans se soucier des changements d'emplacement ou de nom. Les seuls profils qui peuvent être modifiés au moyen de fragments sont les profils par défaut, Invite de commandes et PowerShell, ainsi que les profils dynamiques. La spécification d'un GUID est facultative, mais vivement encouragée.

Les GUID sont générés à l'aide d'un générateur UUID version 5 qui prend en charge l'encodage UTF-16LE sans indicateur d'ordre des octets (BOM).

Le GUID d'espace de noms pour le Terminal Windows dans le cas de profils créés par des plug-ins et des fragments est `{f65ddb7e-706b-4499-8a50-40313caf510a}`. Les profils créés par l'équipe Terminal Windows utilisent un GUID distinct (`{2bde4a90-d05f-401c-9492-e40884ead1d8}`). Cela permet de lever l'ambiguïté entre les profils créés par l'équipe Terminal Windows et ceux créés par des plug-ins ou des fragments afin qu'ils n'entrent jamais en conflit accidentellement.

Comment déterminer le GUID d'un profil existant

Le GUID d'un profil à mettre à jour dépend du type de profil :

Un profil fourni par un tiers et stocké dans un emplacement de fragments Terminal Windows standard nécessite le GUID d'espace de noms de fragment et de profil `{f65ddb7e-706b-4499-8a50-40313caf510a}`, le GUID d'espace de noms d'application et le nom du profil. Pour un fragment de profil nommé « Git Bash » fourni par l'application « Git », le GUID généré est : `{2ece5bfe-50ed-5f3a-ab87-5cd4baafed2b}`.

Un profil généré automatiquement par le Terminal Windows nécessite le GUID interne du Terminal Windows `{2bde4a90-d05f-401c-9492-e40884ead1d8}` et le nom du profil. Pour un profil nommé « Ubuntu » généré automatiquement durant l'installation de WSL, le GUID résultant est : `{2c4de342-38b7-51cf-b940-2309a097f518}`. Contrairement à l'exemple de fragment précédent, aucun « nom d'application » n'est impliqué.

Génération d'un nouveau GUID de profil

Pour générer un GUID pour un profil entièrement nouveau avant de le distribuer, vous pouvez utiliser l'exemple Python 3 suivant. Il génère un GUID basé sur le GUID d'espace de noms de fragment et de profil pour un profil appelé « Git Bash » stocké dans un

dossier de fragments Terminal Windows standard sous le nom d'application « Git », ce qui correspond précisément au contrôle d'intégrité.

Python

```python
import uuid

# The Windows Terminal namespace GUID for custom profiles & fragments
terminalNamespaceGUID = uuid.UUID("{f65ddb7e-706b-4499-8a50-40313caf510a}")

# The Application Namespace GUID
appNamespaceGUID = uuid.uuid5(terminalNamespaceGUID, "Git".encode("UTF-16LE").decode("ASCII"))

# Calculate the example GUID for the 'Git Bash' profile
profileGUID = uuid.uuid5(appNamespaceGUID, "Git Bash".encode("UTF-16LE").decode("ASCII"))

# Output the GUID as Windows Terminal expects it (enclosed in curly brackets)
print(f"{{{profileGUID}}}")
```

Calcul d'un GUID pour un profil intégré

Pour calculer un GUID pour un profil intégré, comme un profil WSL généré automatiquement, vous pouvez utiliser l'exemple Python 3 suivant. Il calcule un GUID basé sur le GUID d'espace de noms Terminal Windows pour un profil appelé « Ubuntu » qui a été généré automatiquement pour la distribution WSL, ce qui correspond précisément au contrôle d'intégrité.

Python

```python
import uuid

# The Windows Terminal namespace GUID automatically generated profiles
terminalNamespaceGUID = uuid.UUID("{2bde4a90-d05f-401c-9492-e40884ead1d8}")

# Calculate the example GUID for the 'Git Bash' profile
profileGUID = uuid.uuid5(terminalNamespaceGUID, "Ubuntu".encode("UTF-16LE").decode("ASCII"))

# Output the GUID as Windows Terminal expects it (enclosed in curly brackets)
print(f"{{{profileGUID}}}")
```

Configuration minimale requise pour les paramètres ajoutés avec des fragments

Les profils qu'il est possible d'ajouter aux paramètres utilisateur avec des fragments JSON font l'objet de restrictions :

- Un nouveau profil ajouté avec des fragments doit, au minimum, s'attribuer un nom.
- Un nouveau modèle de couleurs ajouté avec des fragments doit s'attribuer un nom et définir chaque couleur de la table des couleurs (de « black » à « brightWhite » dans l'exemple d'image ci-dessus).

Où placer les fichiers de fragments JSON

L'emplacement des fichiers de fragments JSON varie en fonction de la méthode d'installation de l'application qui souhaite placer ces fichiers.

Applications Microsoft Store

Pour les applications installées par le biais du Microsoft Store (ou similaire), l'application doit se déclarer comme extension d'application. Découvrez-en plus sur la création et l'hébergement d'une extension d'application. La section nécessaire est répliquée ici. Le fichier appxmanifest du package doit inclure :

XML

```
<Package
  ...
  xmlns:uap3="http://schemas.microsoft.com/appx/manifest/uap/windows10/3"
  IgnorableNamespaces="uap uap3 mp">
  ...
    <Applications>
      <Application Id="App" ... >
        ...
        <Extensions>
          ...
          <uap3:Extension Category="windows.appExtension">
            <uap3:AppExtension
Name="com.microsoft.windows.terminal.settings"
                               Id="<id>"
                               PublicFolder="Public">
            </uap3:AppExtension>
          </uap3:Extension>
        </Extensions>
      </Application>
    </Applications>
```

```
    ...
</Package>
```

Points clés à prendre en compte :

- Le champ `"Name"` doit être `com.microsoft.windows.terminal.settings` pour que le Terminal Windows puisse détecter l'extension.
- Le champ `"Id"` peut être rempli selon les préférences du développeur.
- Le champ `"PublicFolder"` doit contenir le nom du dossier, relatif à la racine du package, où les fichiers JSON sont stockés (ce dossier est généralement appelé « Public », mais peut être renommé par le développeur).
- Dans le dossier Public, les fichiers JSON doivent être stockés dans un sous-répertoire appelé « Fragments ».

Applications installées à partir du web

Pour les applications installées à partir du web, deux cas de figure se présentent.

Dans le premier cas, l'installation est destinée à tous les utilisateurs du système. Les fichiers JSON doivent alors être ajoutés au dossier :

`C:\ProgramData\Microsoft\Windows Terminal\Fragments\{app-name}\{file-name}.json`

Dans le deuxième cas, l'installation est réservée à l'utilisateur actuel. Les fichiers JSON doivent alors être ajoutés au dossier :

`C:\Users\<user>\AppData\Local\Microsoft\Windows Terminal\Fragments\{app-name}\{file-name}.json`

Notez que les dossiers `ProgramData` et `LocalAppData` sont des dossiers connus auxquels le programme d'installation doit avoir accès. Dans les deux cas, si le répertoire `Windows Terminal\Fragments` n'existe pas, le programme d'installation doit le créer. `{app-name}` doit être unique à votre application et `{file-name}.json` peut être ce que vous voulez (le terminal lira tous les fichiers .json dans ce répertoire).

Cascadia Code

Article • 21/03/2023

Cascadia Code est une nouvelle police à espacement fixe de Microsoft qui fournit une expérience actualisée pour les applications en ligne de commande et les éditeurs de texte. Cascadia Code a été développé à côté du Terminal Windows. Il est recommandé d'utiliser cette police avec les applications du terminal et les éditeurs de texte, tels que Visual Studio et Visual Studio Code.

Versions de Cascadia Code

Plusieurs versions de Cascadia Code sont disponibles et incluent des ligatures et des glyphes. Toutes les versions de Cascadia Code peuvent être téléchargées à partir de la page des versions de Cascadia Code GitHub. Le Terminal Windows expédie Cascadia Code et Cascadia Mono dans leur package et utilise Cascadia Mono par défaut.

Nom de la police	Comprend les ligatures	Comprend les glyphes Powerline
Cascadia Code	Oui	Non
Cascadia Mono	Non	Non
Cascadia Code PL	Oui	Oui
Cascadia Mono PL	Non	Oui

Ligatures de programmation et Powerline

Powerline est un plug-in de ligne de commande courant qui vous permet d'afficher des informations supplémentaires dans votre invite. Il utilise quelques glyphes supplémentaires pour afficher correctement ces informations.

Les ligatures de programmation sont des glyphes créés en combinant des caractères. Elles sont particulièrement utiles lors de l'écriture de code. Les variantes « code » incluent des ligatures, tandis que les variantes « Mono » les excluent.

Contribution à Cascadia Code

Cascadia Code est concédé sous licence sous la licence SIL Open Font SIL sur GitHub.

Conseils et astuces pour Terminal Windows

Article • 21/03/2023

Lors du premier lancement

Quand vous installez le Terminal Windows pour la première fois, vous recevrez une invite de Windows PowerShell. Le Terminal Windows est fourni avec les profils Windows PowerShell, Invite de commandes et Azure Cloud Shell par défaut.

En plus de ces profils, le terminal crée automatiquement un profil pour toute distribution du Sous-système Windows pour Linux (WSL) installée. Si vous souhaitez installer des distributions WSL supplémentaires sur votre ordinateur, vous pouvez le faire après l'installation de terminal. Les profils de ces distributions s'afficheront automatiquement au prochain lancement du terminal. Ces profils ont pour icône la mascotte Tux de Linux.

> **Notes**
>
> Si vous le souhaitez, vous pouvez changer l'icône de chaque distribution WSL. Les icônes de distribution spécifiques ne sont pas fournies dans le terminal, mais peuvent être téléchargées et affectées avec les paramètres du terminal.

Afficher les paramètres par défaut

Le Terminal Windows est fourni avec un vaste ensemble de paramètres par défaut, notamment des modèles de couleurs et des raccourcis clavier (désormais appelés « actions personnalisées »). Pour afficher le fichier de paramètres par défaut, maintenez la touche `Alt` enfoncée et cliquez sur le bouton Paramètres à l'intérieur du menu déroulant.

Paramètres de profil par défaut

Le Terminal Windows vous permet d'appliquer un paramètre à chaque profil sans avoir à dupliquer le paramètre pour chaque entrée de profil. Pour cela, ajoutez un paramètre à l'intérieur du tableau « defaults » dans l'objet profiles. Découvrez-en plus sur les

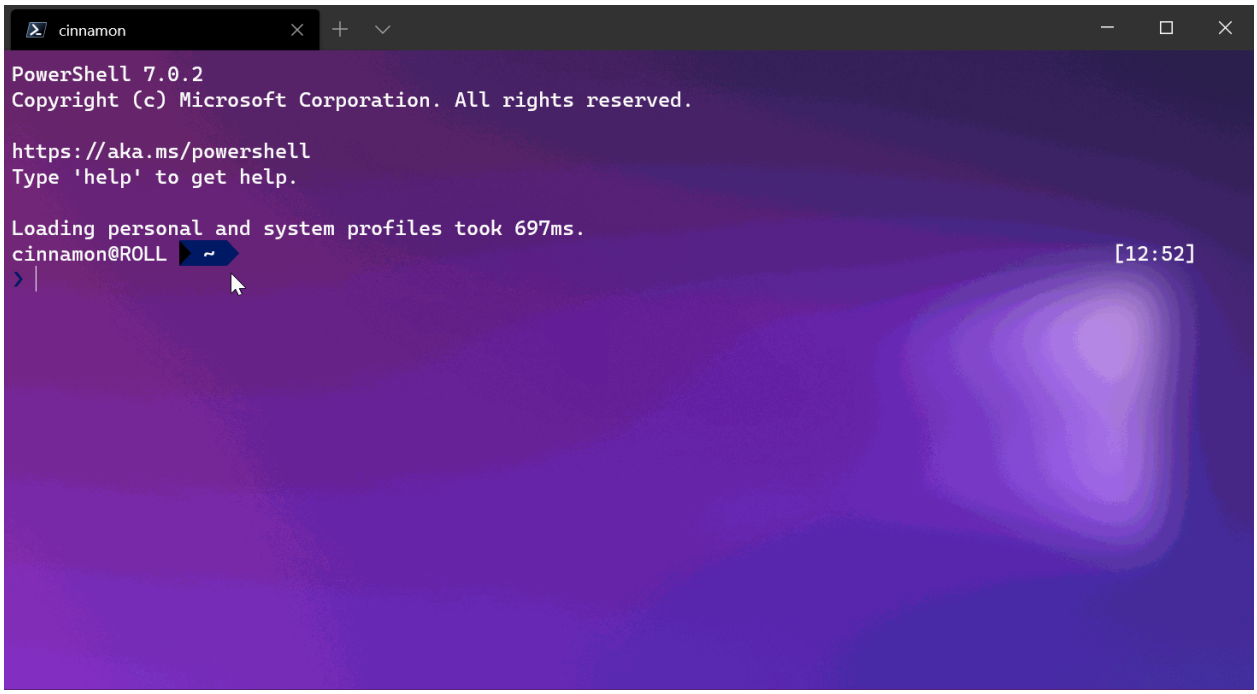
paramètres du profil Général, les paramètres du profil Apparenceet les paramètres du profil Avancé.

JSON

```
"profiles":
    {
        "defaults":
        {
            // Put settings here that you want to apply to all profiles.
            "fontFace": "Cascadia Code"
        },
        "list":
        []
    }
```

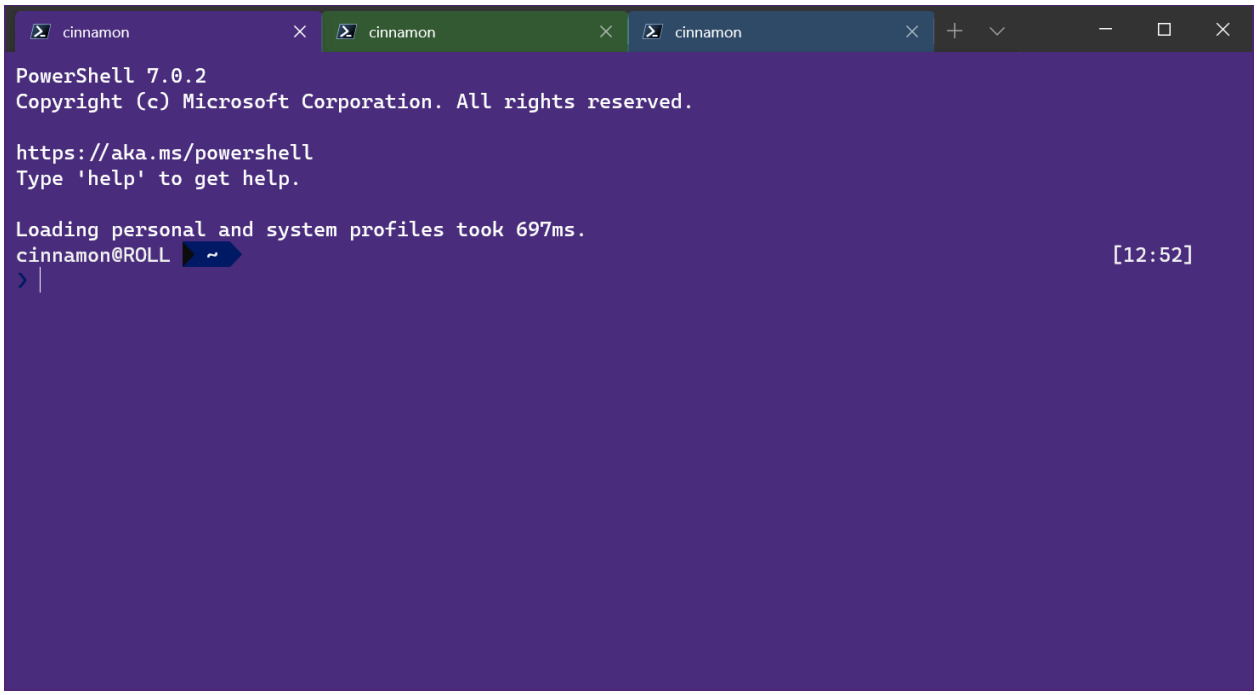
Renommer un onglet

Vous pouvez cliquer avec le bouton droit sur un onglet et sélectionner Renommer un onglet afin de renommer un onglet pour cette session de terminal. Si vous cliquez sur cette option dans le menu contextuel, le titre de l'onglet est changé dans un champ de texte, dans lequel vous pouvez alors modifier le titre. Si vous souhaitez définir le titre de l'onglet pour ce profil pour chaque instance de terminal, vous pouvez en apprendre davantage dans le tutoriel sur les titres des onglets.



Colorer un onglet

Vous pouvez cliquer avec le bouton droit sur un onglet et sélectionner **Couleur...** afin de colorer l'onglet pour cette session de terminal. Vous pouvez faire votre choix parmi une liste prédéfinie de couleurs ou sélectionner **Personnalisé...** pour choisir une couleur à l'aide du sélecteur de couleurs ou des champs RVB/TSV ou hex.



> **Conseil**
>
> Utilisez le champ Hexadécimal afin de définir votre onglet sur la même couleur que la couleur d'arrière-plan pour un aspect épuré.

Il est possible de définir `tabColor` dans le cadre d'un profil. Consultez Profil - Apparence : couleur d'onglet. Par exemple :

JSON

```
{
            "guid": "{1234abc-abcd-1234-12ab-1234abc}",
            "name": "Windows PowerShell",
            "background": "#012456",
            "tabColor": "#012456",
        },
```

Il n'est pas possible de définir `tabColor` dans le cadre d'un modèle de couleurs. Par ailleurs, bien que vous puissiez définir le titre d'un onglet à partir de la ligne de commande avec des séquences d'échappement, il est à ce stade impossible de définir la couleur d'un onglet de cette façon.

Interaction avec la souris

Vous pouvez utiliser une souris pour interagir avec le Terminal Windows de plusieurs façons.

Zoomer avec la souris

Vous pouvez faire un zoom sur la fenêtre de texte du Terminal Windows (pour agrandir ou réduire la taille du texte) en maintenant la touche `Ctrl` enfoncée et en utilisant la roulette de défilement de la souris. Le zoom est conservé pour cette session de terminal. Si vous souhaitez modifier la taille de la police, vous trouverez plus d'informations à ce sujet dans la page Profil - Apparence.

Ajuster l'opacité de l'arrière-plan avec la souris

Vous pouvez ajuster l'opacité de l'arrière-plan en maintenant les touches `Ctrl`+`Maj` enfoncées et en faisant défiler l'écran. L'opacité est conservée pour cette session de terminal. Si vous souhaitez modifier votre opacité d'acrylique pour un profil, vous trouverez plus d'informations sur les effets d'un arrière-plan acrylique dans la page Profil - Apparence.

> ⓘ **Notes**
>
> Dans la **préversion du Terminal Windows** (version 1.12), la modification de l'opacité de l'arrière-plan avec la roulette de la souris utilise par défaut l'opacité de style « vintage », sauf si `useAcrylic` a la valeur true dans vos paramètres. Avant la version 1.12, le terminal utilisait toujours le style « acrylique » pour la transparence.

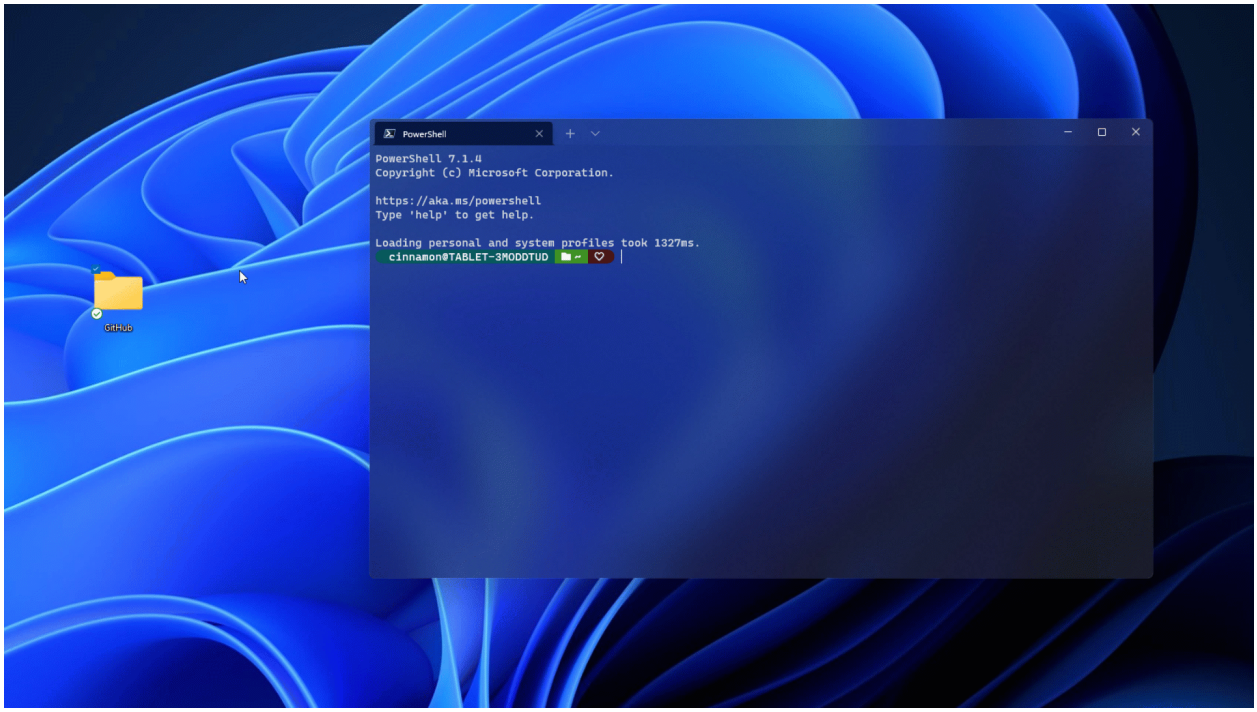
Ouvrir un lien hypertexte

Vous pouvez ouvrir un lien hypertexte à l'intérieur du Terminal Windows en utilisant la combinaison `Ctrl` + clic.

Faire glisser-déposer un fichier ou un dossier à ouvrir

Vous pouvez faire glisser-déposer un fichier ou un dossier sur le bouton Nouvel onglet pour ouvrir votre profil par défaut avec ce fichier ou dossier donné. Par défaut, un nouvel onglet s'ouvre. Vous pouvez soit maintenir la touche `Alt` enfoncée pour ouvrir

un nouveau volet dans votre onglet actuel, soit maintenir la touche `Maj` enfoncée pour ouvrir une nouvelle fenêtre.



Copier/Coller

Vous pouvez cliquer avec le bouton droit de la souris pour copier du texte dans le Presse-papiers et le coller dans le Terminal Windows.

Le Terminal Windows comprend également un paramètre **copyOnSelect** que vous pouvez définir avec la valeur `true` pour copier immédiatement tout texte sélectionné avec la souris dans le Presse-papiers. Dans ce cas, En cliquant sur le bouton droit de votre souris, vous effectuez toujours un collage.

Prise en charge de la souris dans les terminaux virtuels et WSL

Le Terminal Windows prend en charge les entrées à la souris dans les applications Sous-système Windows pour Linux (WSL) ainsi que les applications Windows qui utilisent des entrées par terminal virtuel (VT). Cela signifie que des applications telles que tmux et Midnight Commander savent quand vous sélectionnez des éléments dans la fenêtre du terminal. Si une application est en mode souris, vous pouvez maintenir la touche `Maj` enfoncée pour effectuer une sélection au lieu d'envoyer une entrée VT.

Envoyer des commandes d'entrée avec une combinaison de touches

Le Terminal Windows vous permet d'envoyer des entrées à votre shell avec une combinaison de touches. Pour cela, vous pouvez utiliser la structure suivante dans le tableau `"actions"` de votre fichier settings.json.

JSON

```
{ "command": {"action": "sendInput", "input": ""}, "keys": "" }
```

Vous pouvez également ajouter une valeur `"name": ""` si vous le souhaitez.

Effacer l'écran

L'utilisation d'un raccourci clavier pour envoyer une entrée au shell peut être utile pour les commandes que vous exécutez souvent. Voici un exemple montrant comment effacer votre écran :

JSON

```
{ "command": {"action": "sendInput", "input": "clear\r"}, "keys": "alt+k",
"name": "clear terminal" }
```

Naviguer vers le répertoire parent

La navigation vers le répertoire parent avec une combinaison de touches peut également être utile.

JSON

```
{ "command": {"action": "sendInput", "input": "cd ..\r"}, "keys": "ctrl+up"
}
```

Vous pouvez également utiliser cette fonctionnalité pour exécuter des builds ou des scripts de test.

Mode focus

Le « mode Focus » masque la barre de titre et les onglets normalement situés en haut de Terminal Windows, ce qui vous permet de vous concentrer uniquement sur le contenu du terminal. Il est similaire au « mode Zen » de Visual Studio Code.

Pour entrer en mode Focus, ouvrez la palette de commandes à l'aide de `Ctrl` + `Shift` + `p`, accédez au « mode Focus », puis sélectionnez « Activer/désactiver le mode Focus ». Pour quitter le mode Focus, répétez les mêmes étapes.

Pour définir le lancement du mode focus à chaque démarrage du terminal Windows, ouvrez les **Paramètres** (`Ctrl` + `,`), puis sélectionnez l'onglet **Démarrage**. Sous **Mode de lancement**, sélectionnez **Focus** (ou **Focus maximal**, qui correspond au mode Focus avec l'application Terminal Windows agrandie). Sélectionnez **Enregistrer** avant de quitter. La prochaine fois que vous lancerez Terminal Windows, il s'ouvrira en mode Focus. Pour empêcher le lancement de Terminal Windows en mode Focus, suivez les mêmes étapes, mais sélectionnez **Par défaut** dans la liste des options de **Mode de lancement**.

Pour ajouter une touche de raccourci (ou combinaison de touches) permettant d'entrer en mode Focus, ouvrez le fichier `settings.json` (`Ctrl` + `Shift` + `,`). Dans votre fichier `settings.json`, recherchez la section "actions":, puis ajoutez la commande suivante :

```JSON
{ "command": "toggleFocusMode", "keys": "ctrl+f12" }
```

Remplacez « ctrl+f12 » par le raccourci/la combinaison de touches de votre choix, mais veillez à ne répéter aucune combinaison de touches existante de la liste Actions. Vous pouvez également voir la liste des actions avec les combinaisons de touches associées ainsi que **+ Ajouter de nouvelles liaisons** sous l'onglet **Actions** du tableau de bord **Paramètres** de terminal Windows. N'oubliez pas d'**enregistrer** après avoir apporté des changements. Vous pouvez désormais activer/désactiver le mode Focus à l'aide de la touche de raccourci « action » que vous avez créée. (Dans le cas de notre exemple, `Shift` + `F12`).

Pour en savoir plus sur cette commande, consultez toggleFocusMode.

Mode Quake

Le « mode Quake » est le nom du mode spécial dans lequel se retrouve le terminal lorsqu'une fenêtre est nommée `_quake`. Quand une fenêtre est en mode Quake :

- Le terminal est automatiquement ancré à la moitié supérieure du moniteur.
- La fenêtre ne peut plus être redimensionnée horizontalement ou par le haut. Elle ne peut être redimensionnée que par le bas.

- La fenêtre passe automatiquement en mode Focus (notez que plusieurs onglets peuvent être en mode Focus).
- Quand windowingBehavior a la valeur `"useExisting"` ou `"useAnyExisting"`, l'existence de la fenêtre `_quake` est ignorée.
- Quand elle est réduite, la fenêtre est masquée dans la barre des tâches et avec les touches `Alt`+`Tab`.
- Une seule fenêtre à la fois peut être en mode Quake.

Vous pouvez créer une fenêtre en mode Quake en liant l'action `quakeMode` ou en exécutant manuellement la ligne de commande suivante :

Console

```
wt -w _quake
```

> ⓘ **Notes**
>
> Si aucune action **quakeMode** n'est liée et que vous réduisez la fenêtre Quake, vous devez accéder au gestionnaire des tâches pour quitter cette fenêtre de terminal.

Tutoriel – Configurer une invite personnalisée pour PowerShell ou WSL avec Oh My Posh

Article • 05/10/2023

Ce tutoriel fournit des ressources et des instructions pour vous aider à personnaliser votre invite de commandes pour PowerShell ou Sous-système Windows pour Linux (WSL) avec Oh My Posh. Oh My Posh fournit une expérience d'invite de commandes entièrement personnalisée qui fournit des invites et des codes de couleurs d'état Git.

Dans ce tutoriel, vous allez découvrir comment :

- ✓ Installer une police Nerd
- ✓ Personnaliser votre invite PowerShell avec Oh My Posh
- ✓ Personnaliser votre invite WSL avec Oh My Posh
- ✓ Utiliser Terminal-Icons pour ajouter des icônes de fichier ou dossier manquantes

Installer une police Nerd

Les invites de commandes personnalisées utilisent souvent des glyphes (symboles graphiques) pour appliquer un style à l'invite. Si votre police n'inclut pas les glyphes appropriés, vous pouvez voir différents caractères de remplacement Unicode « □ » dans votre invite de commandes. Pour voir tous les glyphes dans votre terminal, nous vous recommandons d'installer une police Nerd.

Si vous souhaitez une police ressemblant à Cascadia Code, la police Caskaydia Cove Nerd a été créée à partir du dépôt Cascadia Code par un membre de la communauté.

Une fois le téléchargement terminé, décompressez la police et installez-la sur votre système. (Comment ajouter une nouvelle police à Windows).

Pour définir une police Nerd à utiliser avec Oh My Posh et Terminal-Icons, ouvrez l'interface utilisateur des paramètres du Terminal Windows en sélectionnant **Paramètres** (Ctrl+,) dans le menu déroulant Terminal Windows. Sélectionnez le profil auquel vous voulez appliquer la police, par exemple PowerShell, puis l'onglet **Apparence**. Dans le menu déroulant **Type de police**, sélectionnez *CaskaydiaCove Nerd Font* ou la police Nerd que vous souhaitez utiliser avec votre invite de commandes personnalisée.

> ⓘ **Notes**
>
> Pour utiliser une police de terminal qui ne prend pas en charge les icônes de glyphe, comme **Cascadia Code PL**, vous pouvez envisager d'utiliser un thème Oh My Posh contenant la fonction `minimal`, qui indique qu'aucune icône supplémentaire n'est nécessaire.

Personnaliser votre invite PowerShell avec Oh My Posh

Oh My Posh vous permet d'utiliser un jeu de couleurs complet pour définir et afficher votre invite de commandes. Vous pouvez utiliser des thèmes intégrés ou créer votre propre thème.

Installer Oh My Posh pour PowerShell

Pour personnaliser votre invite PowerShell, vous pouvez installer Oh My Posh en utilisant WinGet. Entrez la commande :

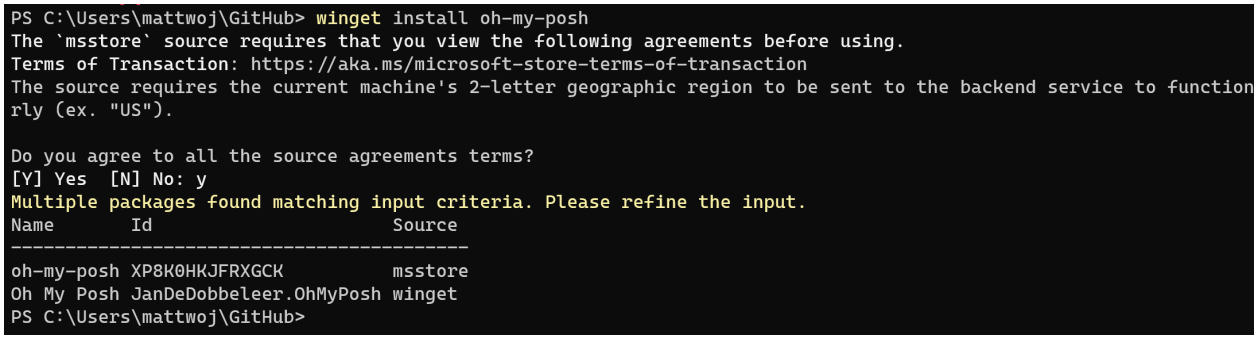
PowerShell

```
winget install JanDeDobbeleer.OhMyPosh
```

Les éléments suivants sont installés :

- `oh-my-posh.exe` : exécutable Windows
- `themes` : Derniers thèmes Oh My Posh

Vous devrez accepter les termes sources et pourrez rencontrer des cas où plusieurs packages sont disponibles. Dans ce cas, sélectionnez l'ID de package que vous souhaitez utiliser et entrez à nouveau la commande : `winget install <package ID>`.



Pour utiliser la version du Microsoft Store de Oh My Posh, qui sera automatiquement mise à jour lorsque de nouvelles versions sont disponibles, utilisez la commande :

PowerShell

```
winget install XP8K0HKJFRXGCK
```

Entrez `oh-my-posh version` pour confirmer le numéro de version de votre installation Oh My Posh. Pour vous assurer que vous disposez des dernières mises à jour, vous pouvez utiliser la commande suivante : `winget upgrade oh-my-posh`.

> ⓘ **Notes**
>
> Si vous souhaitez installer la version la plus récente de Oh My Posh dans PowerShell, vous pouvez d'abord supprimer les fichiers mis en cache du module OMP et désinstaller l'ancien module. Il existe des instructions pour ce faire dans la **documentation Oh My Posh**. Si vous êtes plus familier avec le programme d'installation de **Scoop** ou une méthode d'installation manuelle qui permet l'automatisation, ceux-ci peuvent également être utilisés pour l'installation sur Windows, suivez simplement les instructions de la **documentation Oh My Posh**.

Choisir et appliquer un thème d'invite PowerShell

Vous pouvez parcourir la liste complète des thèmes dans la page Oh My Posh themes.

Choisissez un thème et mettez à jour votre profil PowerShell avec cette commande. (Vous pouvez remplacer `notepad` par l'éditeur de texte de votre choix.)

PowerShell

```
notepad $PROFILE
```

Si vous recevez une erreur de chemin d'accès, vous n'avez peut-être pas encore de profil pour PowerShell. Utilisez la commande PowerShell suivante pour créer un profil, puis réessayez de l'ouvrir avec un éditeur de texte.

PowerShell

```
new-item -type file -path $profile -force
```

Ajoutez le code suivant à la fin de votre fichier de profil PowerShell pour définir le thème `paradox`. (Remplacez `paradox` par le thème de votre choix.)

PowerShell

```
oh-my-posh init pwsh --config "$env:POSH_THEMES_PATH\paradox.omp.json" |
Invoke-Expression
```

À présent, chaque nouvelle instance de PowerShell démarre en important Oh My Posh et en définissant votre thème de ligne de commande.

Si vous recevez une erreur de script lors de la tentative d'ouverture d'une nouvelle instance PowerShell, votre stratégie d'exécution pour PowerShell peut être restreinte. Pour définir une stratégie d'exécution PowerShell sans restriction, vous devez lancer PowerShell en tant qu'administrateur, puis utiliser la commande suivante :

PowerShell

```
Set-ExecutionPolicy -ExecutionPolicy Unrestricted
```

Notes

Il ne s'agit pas de votre profil Terminal Windows. Votre profil PowerShell est un script qui s'exécute chaque fois que PowerShell démarre. **En savoir plus sur les profils PowerShell**.

Conseil

Pour obtenir des réponses aux questions fréquentes ou aux problèmes courants, consultez le **FAQ Oh My Posh**. Pour en savoir plus sur la configuration et les

paramètres généraux, notamment la restauration du répertoire de travail actuel, consultez la **documentation Oh My Posh**.

Personnaliser votre invite WSL avec Oh My Posh

Oh My Posh vous permet à présent de personnaliser des invites WSL, comme vous le feriez pour une invite PowerShell à l'aide de thèmes intégrés.

Installer Oh My Posh pour WSL

Nous vous recommandons d'installer Oh My Posh pour WSL, que vous utilisiez Bash, Zsh ou autre chose, en suivant le guide d'installation Linux dans la documentation Oh My Posh.

Actuellement, le chemin recommandé pour la personnalisation des invites WSL avec Oh My Posh utilise le gestionnaire de package Homebrew pour l'installation. (Homebrew fonctionne maintenant avec WSL) Lors de l'installation de Homebrew pour Linux, veillez à suivre les instructions des étapes suivantes pour ajouter Homebrew à votre CHEMIN et à votre script de profil d'interpréteur de commandes bash.

Homebrew va installer :

- `oh-my-posh` : un executable, ajouté à /usr/local/bin
- `themes` : les derniers thèmes Oh My Posh

Choisir et appliquer un thème d'invite WSL

Les thèmes Oh My Posh se trouvent dans le répertoire oh-my-posh sous forme de fichiers JSON. Pour trouver ce dernier, entrez `cd $(brew --prefix oh-my-posh)`, puis simplement `cd themes` et `ls` pour obtenir la liste. Pour Ubuntu-20.04 exécuté par le biais de WSL, le chemin ressemble probablement à celui-ci : `\\wsl.localhost\Ubuntu-20.04\home\linuxbrew\.linuxbrew\Cellar\oh-my-posh\6.34.1\themes`. Vous pouvez également voir à quoi ressemblent les thèmes dans la documentation Oh My Posh sur les thèmes.

Pour utiliser un thème, copiez-le à partir du dossier `themes` vers votre dossier `$Home`, puis ajoutez cette ligne au bas du fichier `.profile` qui se trouve dans votre dossier `$Home` :

Bash

```
eval "$(oh-my-posh init bash --config ~/jandedobbeleer.omp.json)"
```

Vous pouvez remplacer `jandedobbeleer.omp.json` par le nom du thème que vous préférez utiliser tant qu'il est copié dans votre dossier `$Home`.

Si vous utilisez oh-my-posh à la fois dans Windows avec PowerShell et avec WSL, vous pouvez également partager votre thème PowerShell avec WSL en pointant sur un thème dans le dossier de base de votre utilisateur Windows. Dans le chemin `.profile` de votre distribution WSL, remplacez ~ par le chemin : `/mnt/c/Users/<WINDOWSUSERNAME>`. Remplacement de `<WINDOWSUSERNAME>` par votre propre nom d'utilisateur Windows.

Si vous le souhaitez, vous pouvez personnaliser les thèmes Oh My Posh.

Utiliser Terminal-Icons pour ajouter des icônes de fichier ou dossier manquantes

Terminal-icons est un module PowerShell qui permet d'ajouter des icônes de fichier et dossier susceptibles d'être manquantes lors de l'affichage de fichiers ou dossiers dans Terminal Windows, en recherchant leur icône appropriée en fonction de leur nom ou extension. Le module tente d'utiliser des icônes pour les fichiers/dossiers connus, mais revient à une icône de fichier ou dossier générique si aucune icône n'est trouvée.

Pour installer Terminal-Icons avec PowerShell, utilisez la commande suivante :

PowerShell

```
Install-Module -Name Terminal-Icons -Repository PSGallery
```

Pour plus d'informations, notamment sur l'utilisation et les commandes, consultez le dépôt Terminal-Icons sur GitHub.

Ressources supplémentaires

- Documentation Oh My Posh
- Dépôt Terminal-Icons
- Documentation Posh-Git : Posh-Git est un module PowerShell qui intègre Git et PowerShell en fournissant des informations récapitulatives de l'état Git affichables dans l'invite PowerShell.

- Documentation PowerLine : PowerLine est un plug-in de ligne d'état pour vim qui fournit des lignes d'état et des invites pour plusieurs autres applications, notamment zsh, bash, tmux, IPython, Awesome, i3 et Qtile.

Tutoriel : SSH dans le Terminal Windows

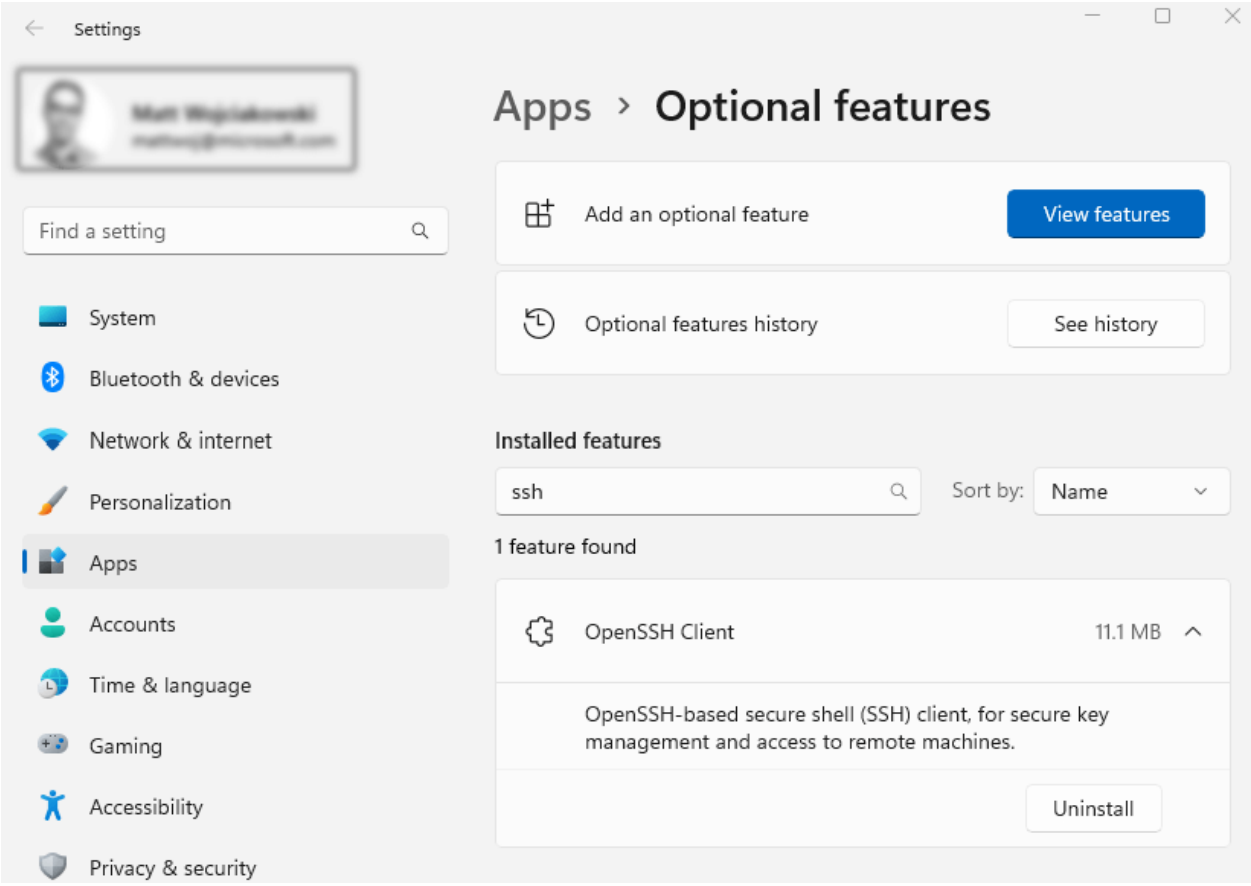
Article • 04/10/2023

Windows a un client SSH intégré que vous pouvez utiliser dans Windows Terminal. Dans ce didacticiel, vous allez apprendre à configurer un profil dans le Terminal Windows qui utilise le protocole SSH. Notez que cette fonctionnalité est en préversion.

Accéder au client SSH Windows

Les builds les plus récentes de Windows 10 et de Windows 11 incluent un serveur SSH intégré et un client basés sur OpenSSH, un outil de connectivité pour la connexion à distance qui utilise le protocole SSH. OpenSSH chiffre tout le trafic entre le client et le serveur de façon à éliminer les écoutes clandestines, le détournement de connexion et d'autres attaques.

Par défaut, le client OpenSSH se trouve dans le répertoire `C:\Windows\System32\OpenSSH`. Vous pouvez également vérifier qu'il est installé en accédant à Paramètres Windows > Applications > Fonctionnalités facultatives, puis en recherchant « OpenSSH » dans vos fonctionnalités installées.



Pour plus d'informations sur la configuration d'OpenSSH, consultez Configuration du serveur OpenSSH pour Windows.

> ⓘ **Notes**
>
> Le Terminal Windows version 1.XX+ peut générer dynamiquement des profils permettant de se connecter aux hôtes SSH au sein de votre **fichier de configuration OpenSSH**.

Créer un profil

Vous pouvez démarrer une session SSH dans votre invite de commandes en exécutant `ssh user@machine` et vous serez invité à entrer votre mot de passe. Pour créer un profil Terminal Windows qui fait cela au démarrage, ajoutez le paramètre `commandline` à un profil dans votre fichier settings.json à l'intérieure de la liste (`list`) d'objets de profil.

JSON

```
{
  "name": "user@machine ssh profile",
  "commandline": "ssh user@machine"
}
```

Pour plus d'informations, consultez les pages suivantes :

- Profil Terminal Windows - Paramètres généraux

Spécifier le répertoire de départ

Pour spécifier le répertoire de départ d'une session SSH appelée par Terminal Windows, vous pouvez utiliser cette commande :

JSON

```
{
  "commandline": "ssh -t bob@foo \"cd /data/bob && exec bash -l\""
}
```

L'indicateur `-t` force l'allocation de pseudo-terminal. Vous pouvez l'utiliser pour exécuter des programmes basés sur des écrans arbitraires sur un ordinateur distant, par exemple lors de l'implémentation des services de menu. Vous devez utiliser des guillemets doubles d'échappement, car les dérivés Bourne shell n'effectuent aucune analyse supplémentaire pour une chaîne entre guillemets simples.

Pour plus d'informations, voir :

- Problème GH : Comment spécifier le répertoire de départ pour une session SSH ?
- StackOverflow : Comment puis-je exécuter ssh directement dans un répertoire particulier ?

Ressources

- Comment activer et utiliser les nouvelles commandes SSH intégrées à Windows 10

Tutoriel : Configurer les titres des onglets dans le Terminal Windows

Article • 21/03/2023

Par défaut, le titre de l'onglet est défini sur le titre du shell. Si un onglet est composé de plusieurs volets, le titre de l'onglet est défini sur celui du volet qui a actuellement le focus. Si vous souhaitez personnaliser ce qui est défini comme titre d'onglet, suivez ce tutoriel.

Dans ce tutoriel, vous allez découvrir comment :

- ✓ Utiliser le paramètre `tabTitle`
- ✓ Définir le titre du shell
- ✓ Utiliser le paramètre `suppressApplicationTitle`

Utiliser le paramètre `tabTitle`

Le paramètre `tabTitle` vous permet de définir le titre de départ d'une nouvelle instance d'un shell. S'il n'est pas défini, la valeur `name` du profil est utilisée à la place. Chaque shell répond à ce paramètre différemment.

Shell	Comportement
PowerShell	Le titre est défini.
Invite de commandes	Le titre est défini. Si une commande est en cours d'exécution, elle est ajoutée temporairement à la fin du titre.
Ubuntu	Le titre est ignoré et est défini à la place sur `user@machine:path`
Debian	Le titre est défini.

> ⓘ **Notes**
>
> Bien que Ubuntu et Debian exécutent tous deux bash, ils présentent des comportements différents. Nous tenons à indiquer ici que différentes distributions peuvent avoir des comportements différents.

Définir le titre du shell

Un shell a un contrôle total sur son propre titre. Toutefois, chaque shell définit son titre différemment.

Shell	Commande
PowerShell	`$Host.UI.RawUI.WindowTitle = "New Title"`
Invite de commandes	`TITLE "New Title"`
bash*	`echo -ne "\033]0;New Title\a"`

Notez que certaines distributions Linux (par exemple Ubuntu) définissent automatiquement leur titre lorsque vous interagissez avec le shell. Si la commande ci-dessus ne fonctionne pas, exécutez la commande suivante :

Bash

```
export PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\
[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
echo -ne '\033]0;New Title\a'
```

Le titre est alors remplacé par « New Title ».

Pour un accès plus facile, ajoutez ceci à la fin de `~/.bashrc` :

Bash

```
settitle () {
  export PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\
[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
  echo -ne '\033]0;'"$1"'\a'
}
```

Après avoir rouvert votre shell, vous pouvez maintenant modifier son titre à tout moment en utilisant la commande suivante :

Bash

```
settitle 'New Title'
```

Utiliser le paramètre `suppressApplicationTitle`

Étant donné qu'un shell contrôle son titre, il peut choisir de remplacer le titre de l'onglet à tout moment. Par exemple, le module `posh-git` pour PowerShell ajoute les

informations relatives à votre référentiel Git au titre.

Le Terminal Windows vous permet de supprimer les modifications apportées au titre en définissant `suppressApplicationTitle` sur `true` dans votre profil. Ainsi, les nouvelles instances du profil définissent votre titre visible sur `tabTitle`. Si `tabTitle` n'est pas défini, le titre visible est défini sur la valeur `name` du profil.

Notez que cela dissocie le titre du shell du titre visible présenté sur l'onglet. Si vous lisez la variable du shell où le titre est défini, elle peut être différente du titre de l'onglet.

Ressources

- Définition du titre de la console comme répertoire de travail actuel
- Modifier le titre d'un terminal sur Ubuntu 16.04

Tutoriel : Ouverture d'un onglet ou d'un volet dans le même répertoire au sein de Terminal Windows

Article • 21/03/2023

En règle générale, les actions « nouvel onglet » et « volet fractionné » ouvrent toujours un nouvel onglet/volet dans le `startingDirectory` de ce profil. Toutefois, sur d'autres plateformes, il est courant que les nouveaux onglets utilisent automatiquement le répertoire de travail de l'onglet actuel comme répertoire de départ pour un nouvel onglet. Cela permet à l'utilisateur d'effectuer rapidement plusieurs tâches dans un même répertoire.

Malheureusement, sur Windows, il est difficile de déterminer le répertoire de travail actuel (« CWD ») d'un processus. Même si nous pouvons le rechercher, toutes les applications ne définissent pas réellement leur CWD pendant la navigation. En particulier, Windows PowerShell ne change pas son CWD quand vous utilisez `cd` dans le système de fichiers ! La duplication automatique du CWD de PowerShell serait presque toujours incorrecte.

Heureusement, il existe une solution de contournement. Les applications peuvent envoyer une séquence d'échappement spéciale pour indiquer manuellement au terminal ce que doit être le CWD.

Dans ce tutoriel, vous allez apprendre à :

- ✓ Configurer l'interpréteur de commandes pour indiquer au terminal son répertoire de travail actif
- ✓ Utiliser l'action `duplicateTab` pour ouvrir un onglet avec le même CWD
- ✓ Utiliser l'action `splitPane` pour ouvrir un volet avec le même CWD
- ✓ Utilisation du menu contextuel de l'onglet pour ouvrir des onglets ou des volets avec le même CWD

Configurer votre interpréteur de commandes

Pour indiquer le CWD au terminal, vous devez modifier votre interpréteur de commandes pour qu'il envoie une séquence d'échappement quand vous naviguez dans le système d'exploitation. Heureusement, la plupart des interpréteurs de commandes ont un mécanisme pour configurer l'« invite », qui est exécuté après chaque commande. C'est l'endroit idéal pour ajouter une sortie de ce type.

Windows

Invite de commandes : `cmd.exe`

`cmd` utilise la variable d'environnement `%PROMPT%` pour configurer l'invite de commandes. Vous pouvez facilement ajouter l'invite avec la commande pour définir CWD en utilisant la commande suivante :

Invite de commandes Windows

```
set PROMPT=$e]9;9;$P$e\%PROMPT%
```

Cela ajoute `$e]9;9;$P$e\` à votre invite actuelle. Quand cmd évalue cette invite, il remplace

- `$e` par le caractère d'échappement
- `$p` par le répertoire de travail actuel

Notez que la commande ci-dessus fonctionne seulement pour la session `cmd.exe` actuelle. Pour définir la valeur de manière permanente, APRÈS avoir exécuté la commande ci-dessus, vous pouvez exécuter

Invite de commandes Windows

```
setx PROMPT "%PROMPT%"
```

PowerShell : `powershell.exe` ou `pwsh.exe`

Si vous n'avez jamais changé votre invite PowerShell auparavant, vous devez d'abord consulter about_Prompts.

Ajoutez ce qui suit à votre profil PowerShell :

PowerShell

```
function prompt {
  $loc = $executionContext.SessionState.Path.CurrentLocation;

  $out = "PS $loc$('>' * ($nestedPromptLevel + 1)) ";
  if ($loc.Provider.Name -eq "FileSystem") {
    $out += "$([char]27)]9;9;`"$($loc.ProviderPath)`"$([char]27)\"
  }
  return $out
}
```

PowerShell avec posh-git

Si vous utilisez posh-git, l'invite est déjà modifiée. Dans ce cas, vous ajoutez seulement la sortie nécessaire à l'invite déjà modifiée. L'exemple suivant est une version légèrement modifiée de cet exemple dans la documentation ConEmu :

PowerShell

```
function prompt
{
  $loc = Get-Location

  $prompt = & $GitPromptScriptBlock

  $prompt += "$([char]27)]9;12$([char]7)"
  if ($loc.Provider.Name -eq "FileSystem")
  {
    $prompt += "$([char]27)]9;9;`"$($loc.ProviderPath)`"$([char]27)\"
  }

  $prompt
}
```

PowerShell avec Starship

Si vous utilisez Starship, l'invite est déjà modifiée. Dans ce cas, vous ajoutez seulement la sortie nécessaire à l'invite déjà modifiée.

PowerShell

```
function Invoke-Starship-PreCommand {
  $loc = $executionContext.SessionState.Path.CurrentLocation;
  $prompt = "$([char]27)]9;12$([char]7)"
  if ($loc.Provider.Name -eq "FileSystem")
  {
    $prompt += "$([char]27)]9;9;`"$($loc.ProviderPath)`"$([char]27)\"
  }
  $host.ui.Write($prompt)
}
```

WSL

Les distributions du Sous-système Windows pour Linux utilisent principalement BASH comme interpréteur de commandes.

`bash`

Ajoutez la ligne suivante à la fin de votre fichier de configuration `.bash_profile` :

Bash

```
PROMPT_COMMAND=${PROMPT_COMMAND:+"$PROMPT_COMMAND; "}'printf "\e]9;9;%s\e\\" "$(wslpath -w "$PWD")"'
```

La variable `PROMPT_COMMAND` dans bash indique à bash quelle commande exécuter avant d'afficher l'invite. Nous utilisons l'instruction `printf` pour ajouter la séquence qui définit le répertoire de travail avec le terminal. Le bit `$(wslpath -w "$PWD")` appelle l'exécutable `wslpath` pour convertir le répertoire actuel en chemin de type Windows. Le bit `${PROMPT_COMMAND:+"$PROMPT_COMMAND; "}` est une commande magique bash qui garantit que nous ajoutons cette commande à toutes les commandes existantes (si vous avez déjà défini `PROMPT_COMMAND` ailleurs).

`zsh`

Ajoutez les lignes suivantes à la fin de votre fichier `.zshrc` :

zsh

```
keep_current_path() {
  printf "\e]9;9;%s\e\\" "$(wslpath -w "$PWD")"
}
precmd_functions+=(keep_current_path)
```

Le hook `precmd_functions` indique à zsh quelles commandes exécuter avant d'afficher l'invite. Nous utilisons l'instruction `printf` pour ajouter la séquence qui définit le répertoire de travail avec le terminal. Le bit `$(wslpath -w "$PWD")` appelle l'exécutable `wslpath` pour convertir le répertoire actuel en chemin de type Windows. À l'aide de `precmd_functions+=` assurez-vous que nous ajoutons la fonction `keep_current_path` à toute fonction existante déjà définie pour ce crochet.

Fish

Si vous utilisez l'interpréteur de commandes Fish, ajoutez les lignes suivantes à la fin de votre fichier config situé sur `~/.config/fish/config.fish` :

Bash

```
function storePathForWindowsTerminal --on-variable PWD
    if test -n "$WT_SESSION"
      printf "\e]9;9;%s\e\\" (wslpath -w "$PWD")
    end
end
```

Cette fonction est appelée chaque fois que le chemin actuel change pour vérifier que la session actuelle est ouverte par l'application Terminal Windows (vérification de $WT_SESSION), et qu'elle envoie la commande de système d'exploitation (OSC 9;9;) avec le chemin équivalent Windows (`wslpath -w`) du chemin actuel.

MINGW

Pour MINGW, Git Bash et Cygwin, vous devez modifier `PROMPT_COMMAND`. Pour WSL, remplacez `wslpath` par `cygpath`.

Ajoutez la ligne suivante à la fin de votre fichier `.bashrc` :

Bash

```
PROMPT_COMMAND=${PROMPT_COMMAND:+"$PROMPT_COMMAND; "}'printf "\e]9;9;%s\e\\"
"`cygpath -w "$PWD"`"'
```

> ⓘ **Notes**
>
> Vous ne voyez pas votre interpréteur de commandes favori ici ? N'hésitez pas à ouvrir une **demande de tirage** afin de contribuer à l'ajout d'une solution pour votre interpréteur de commandes préféré.

Utilisation d'actions pour dupliquer le chemin

Une fois que vous avez configuré l'interpréteur de commandes pour indiquer au terminal le répertoire actuel, vous ouvrez facilement un nouvel onglet ou un nouveau volet avec ce chemin.

Ouvrir un nouvel onglet avec `duplicateTab`

Pour ouvrir un nouvel onglet avec le même chemin (et profil) que celui du terminal actuellement actif, utilisez l'action « Dupliquer l'onglet ». Elle est liée par défaut à `Ctrl+Maj+d`, de la façon suivante :

JSON

```
        { "command": "duplicateTab", "keys": "ctrl+shift+d" },
```

(Voir duplicateTab) pour plus d'informations.

Ouvrir un nouveau volet avec `splitPane`

Pour ouvrir un nouveau volet avec le même chemin (et le même profil) que celui du terminal actuellement actif, utilisez l'action « Dupliquer le volet ». Elle n'est **PAS** liée par défaut. La forme la plus simple de cette action est :

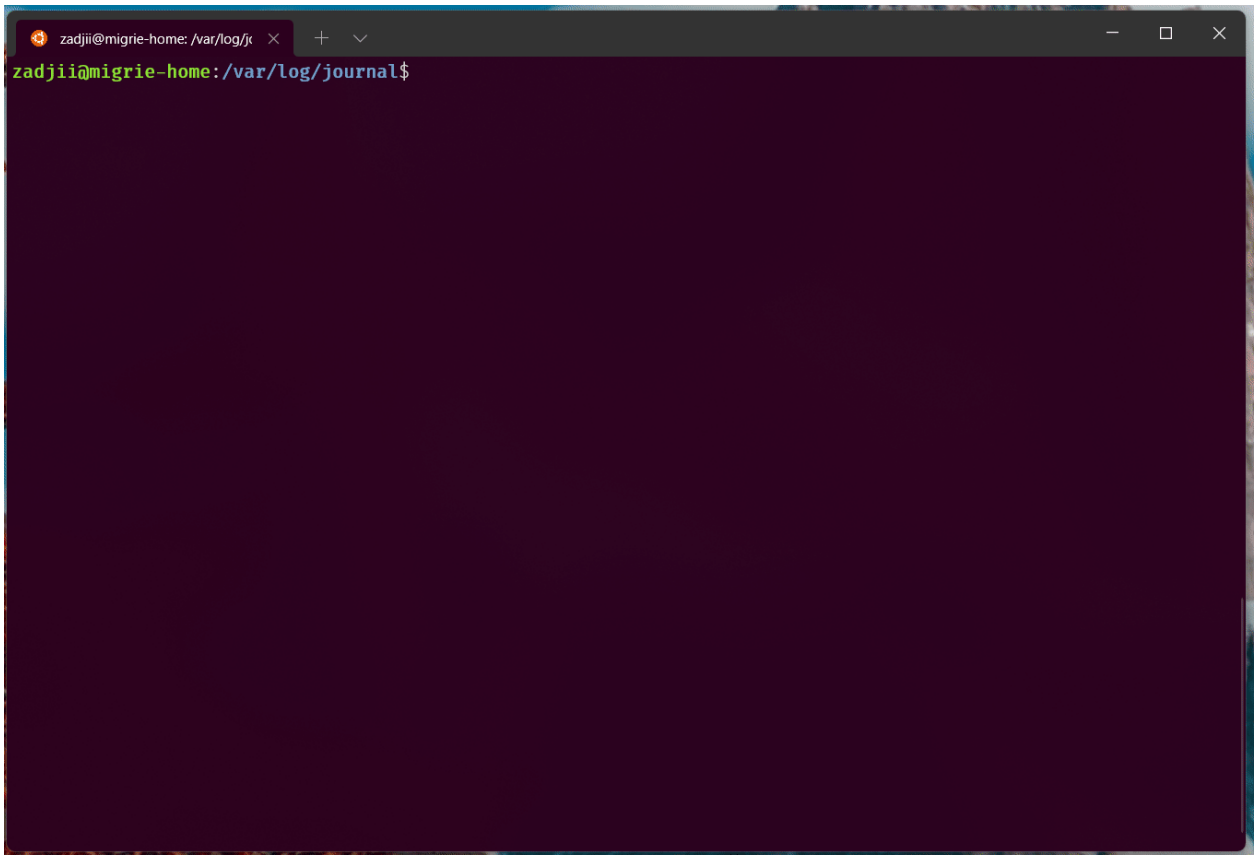
JSON

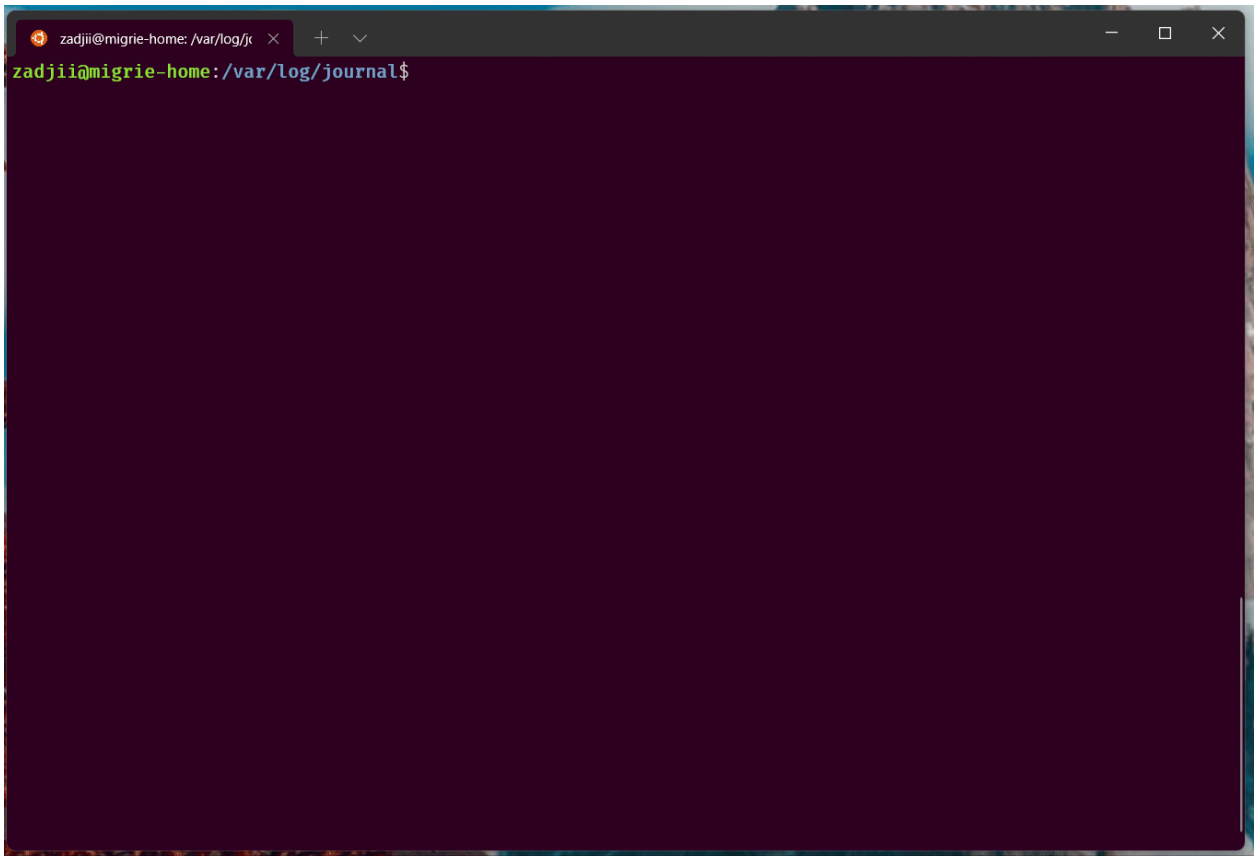
```
        { "command": { "action": "splitPane", "splitMode": "duplicate" } },
```

(Voir splitPane) pour plus d'informations.

Utilisation du menu pour dupliquer le chemin

les actions ci-dessus sont également disponibles dans le menu contextuel de l'onglet, sous les entrées « Dupliquer l'onglet » et « Fractionner le volet ».


zadjii@migrie-home: /var/log/jo
zadjii@migrie-home:/var/log/journal$


zadjii@migrie-home: /var/log/jo
zadjii@migrie-home:/var/log/journal$

Guide sur le Terminal personnalisé

Article • 11/10/2023

Voici quelques modèles de couleurs que vous pouvez essayer ou utiliser comme points de départ pour vos propres conceptions.

Installation de modèles de couleurs

Copiez le code JSON de la section "**schemes**" dans la section appropriée de settings.json. Par exemple :

Avant :

JSON

```
"schemes": [],
```

Après :

JSON

```
 "schemes": [
            {
                "name": "Retro",
                "background": "#000000",
                "black": "#00ff00",
                "blue": "#00ff00",
                "brightBlack": "#00ff00",
                "brightBlue": "#00ff00",
                "brightCyan": "#00ff00",
                "brightGreen": "#00ff00",
                "brightPurple": "#00ff00",
                "brightRed": "#00ff00",
                "brightWhite": "#00ff00",
                "brightYellow": "#00ff00",
                "cyan": "#00ff00",
                "foreground": "#00ff00",
                "green": "#00ff00",
                "purple": "#00ff00",
                "red": "#00ff00",
                "white": "#00ff00",
                "yellow": "#00ff00"
            }
        ]
```

Ajoutez ensuite la section propre au profil, par exemple :

Avant :

JSON

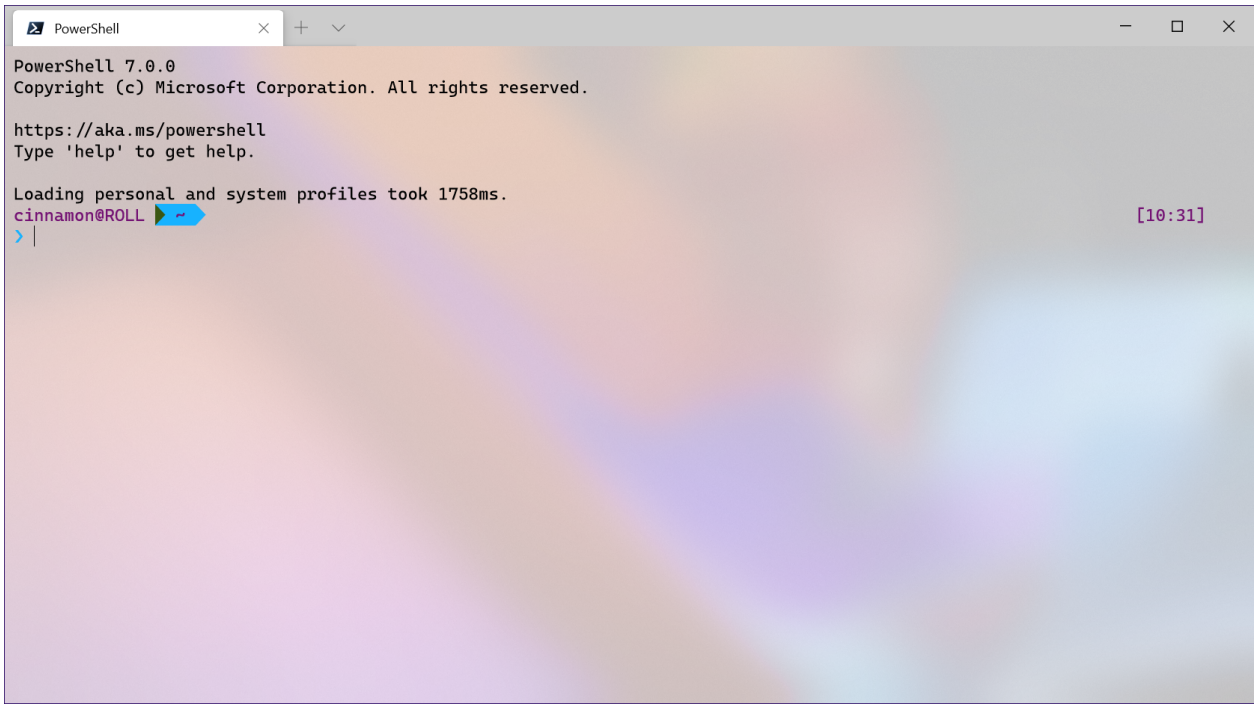
```
{
    "guid": "{234ab24f-34dd-ff3-ade434aad345}",
    "name": "Command Prompt",
    "commandline": "cmd.exe",
    "hidden": false
}
```

Après :

JSON

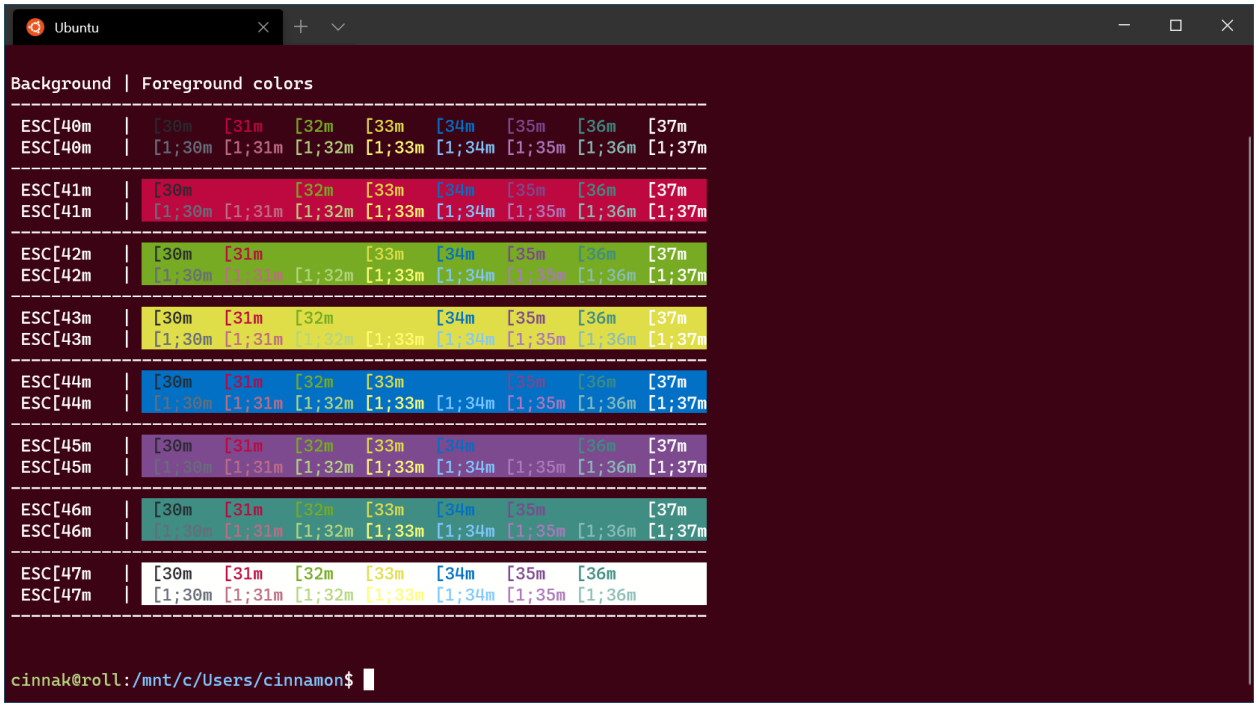
```
{
    "guid": "{234ab24f-34dd-ff3-ade434aad345}",
    "name": "Command Prompt",
    "commandline": "cmd.exe",
    "hidden": false,
    "colorScheme" : "Retro",
    "cursorColor" : "#FFFFFF",
    "cursorShape": "filledBox",
    "fontSize" : 16,
    "padding" : "5, 5, 5, 5",
    "tabTitle" : "Command Prompt",
    "fontFace": "PxPlus IBM VGA8",
    "experimental.retroTerminalEffect": true
}
```

Verre givré



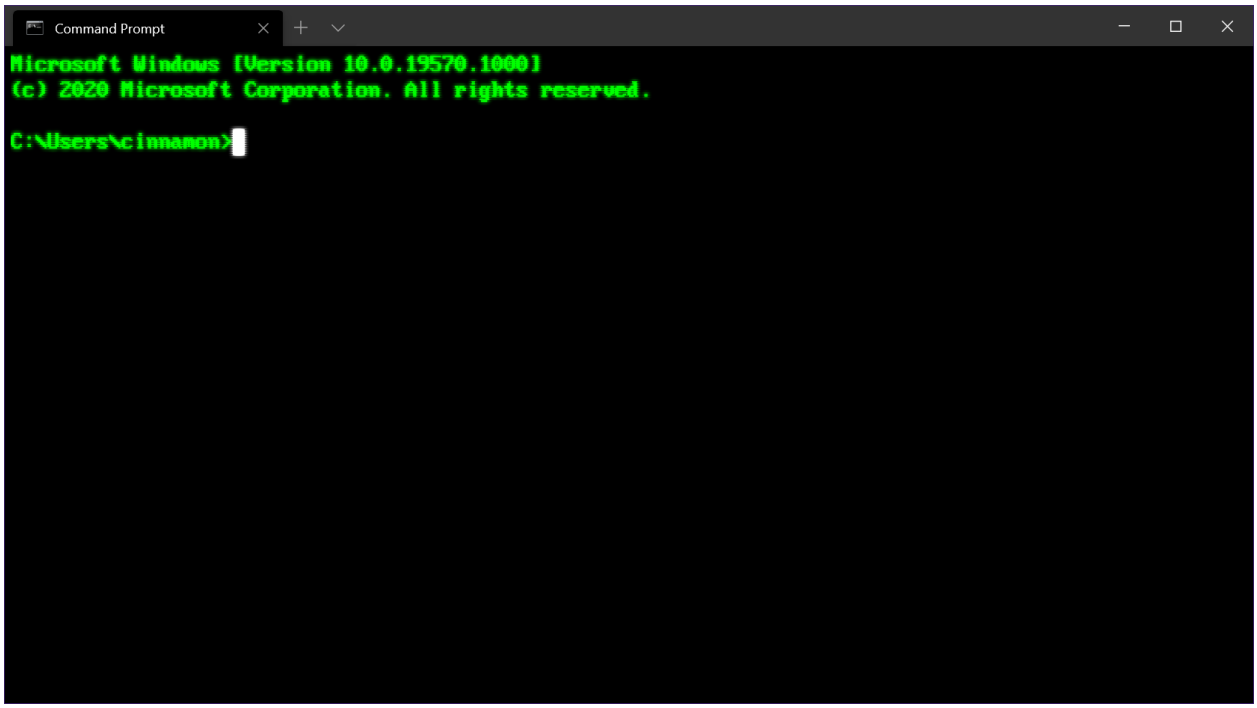
Détails

Raspberry Ubuntu



Détails

Commande rétro



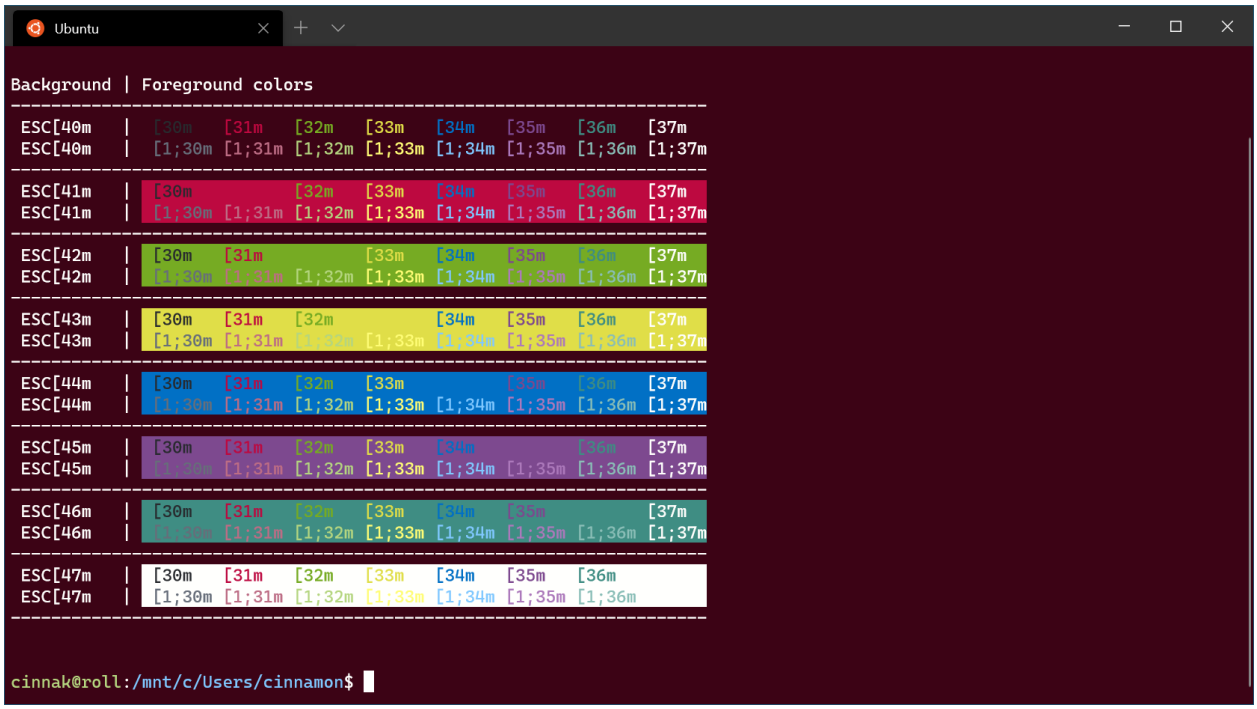
Détails

Partage

Avez-vous un modèle de couleurs Terminal Windows que vous aimeriez partager ? Montrez-le-nous sur Twitter.

Raspberry Ubuntu dans Terminal Windows

Article • 04/02/2024



JSON

```
{
    "theme": "dark",
    "profiles": [
        {
            "name" : "Ubuntu",
            "source" : "Windows.Terminal.Wsl",
            "colorScheme" : "Raspberry",
            "cursorColor" : "#FFFFFF",
            "fontFace" : "Cascadia Code",
            "padding" : "5, 5, 5, 5",
            "suppressApplicationTitle": true,
            "tabTitle": "Ubuntu"
        }
    ],
    "schemes": [
        {
            "name" : "Raspberry",
            "background" : "#3C0315",
            "black" : "#282A2E",
            "blue" : "#0170C5",
            "brightBlack" : "#676E7A",
            "brightBlue" : "#80c8ff",
            "brightCyan" : "#8ABEB7",
            "brightGreen" : "#B5D680",
            "brightPurple" : "#AC79BB",
```

```
                "brightRed" : "#BD6D85",
                "brightWhite" : "#FFFFFD",
                "brightYellow" : "#FFFD76",
                "cyan" : "#3F8D83",
                "foreground" : "#FFFFFD",
                "green" : "#76AB23",
                "purple" : "#7D498F",
                "red" : "#BD0940",
                "white" : "#FFFFFD",
                "yellow" : "#E0DE48"
            }
        ]
    }
```

Collaborer avec nous sur GitHub

La source de ce contenu se trouve sur GitHub, où vous pouvez également créer et examiner les problèmes et les demandes de tirage. Pour plus d'informations, consultez notre guide du contributeur.

Commentaires sur Windows Terminal

Windows Terminal est un projet open source. Sélectionnez un lien pour fournir des commentaires :

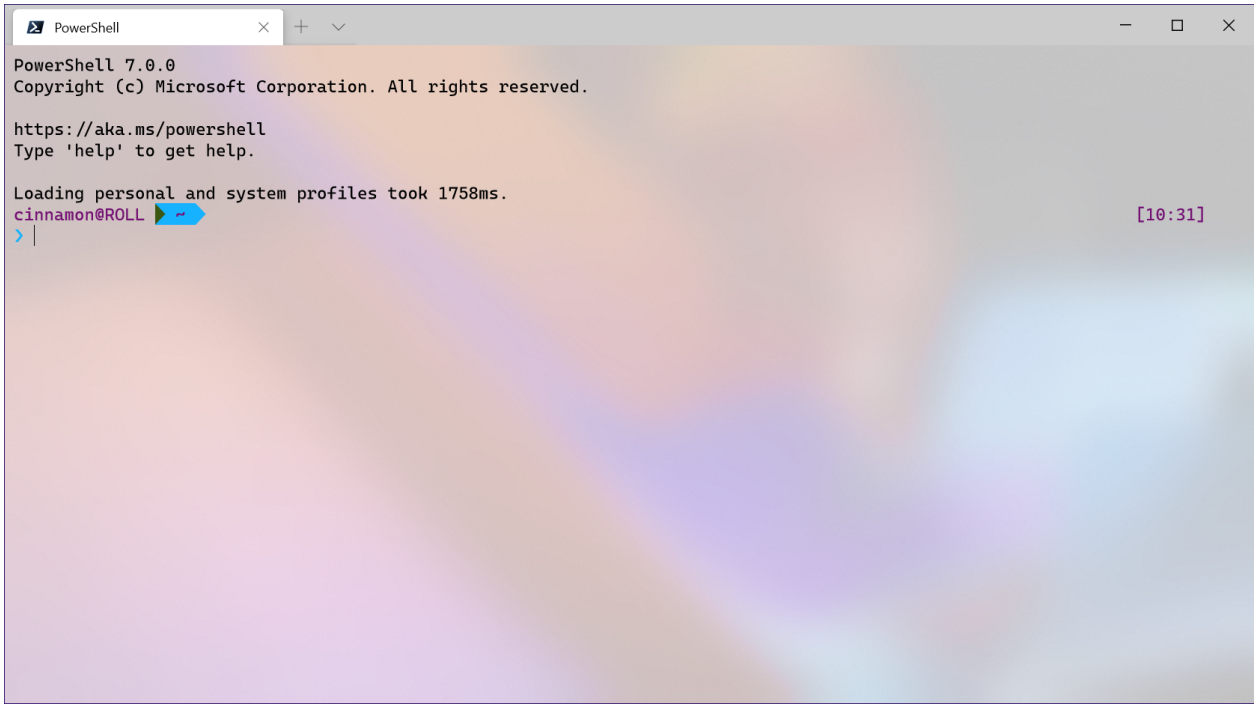
Ouvrir un problème de documentation

Indiquer des commentaires sur le produit

Thème de verre givré dans le Terminal Windows

Article • 09/10/2023

L'invite est stylisée à l'aide de Powerline et utilise la `Cascadia Code PL` police, qui peut être téléchargée à partir de la page des versions GitHub du code Cascadia.



JSON

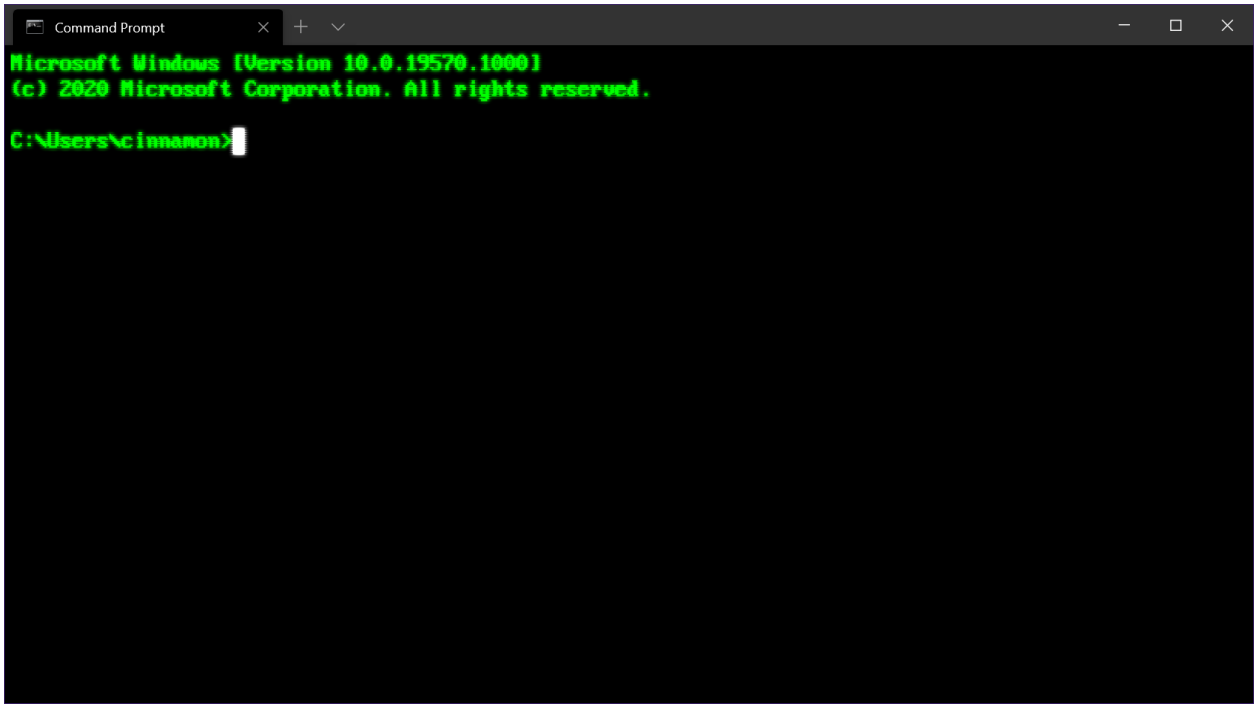
```
    {
        "theme": "light",
        "profiles": [
            {
                "name" : "PowerShell",
                "source" : "Windows.Terminal.PowershellCore",
                "acrylicOpacity": 0.7,
                "colorScheme" : "Frost",
                "cursorColor" : "#000000",
                "fontFace" : "Cascadia Code PL",
                "useAcrylic": true
            }
        ],
        "schemes": [
            {
                "name" : "Frost",
                "background" : "#FFFFFF",
                "black" : "#3C5712",
                "blue" : "#17b2ff",
                "brightBlack" : "#749B36",
                "brightBlue" : "#27B2F6",
```

```
                "brightCyan" : "#13A8C0",
                "brightGreen" : "#89AF50",
                "brightPurple" : "#F2A20A",
                "brightRed" : "#F49B36",
                "brightWhite" : "#741274",
                "brightYellow" : "#991070",
                "cyan" : "#3C96A6",
                "foreground" : "#000000",
                "green" : "#6AAE08",
                "purple" : "#991070",
                "red" : "#8D0C0C",
                "white" : "#6E386E",
                "yellow" : "#991070"
            }
        ]
    }
```

Invite de commandes retro dans le terminal Windows

Article • 07/10/2023

L'invite utilise la police `PxPlus IBM VGA8`, qui n'est pas intégrée au terminal Windows.



JSON

```
    {
        "theme": "dark",
        "profiles": [
            {
                "name": "Command Prompt",
                "commandline": "cmd.exe",
                "closeOnExit" : true,
                "colorScheme" : "Retro",
                "cursorColor" : "#FFFFFF",
                "cursorShape": "filledBox",
                "fontSize" : 16,
                "padding" : "5, 5, 5, 5",
                "tabTitle" : "Command Prompt",
                "fontFace": "PxPlus IBM VGA8",
                "experimental.retroTerminalEffect": true
            }
        ],
        "schemes": [
            {
                "name": "Retro",
                "background": "#000000",
                "black": "#00ff00",
                "blue": "#00ff00",
```

```
                "brightBlack": "#00ff00",
                "brightBlue": "#00ff00",
                "brightCyan": "#00ff00",
                "brightGreen": "#00ff00",
                "brightPurple": "#00ff00",
                "brightRed": "#00ff00",
                "brightWhite": "#00ff00",
                "brightYellow": "#00ff00",
                "cyan": "#00ff00",
                "foreground": "#00ff00",
                "green": "#00ff00",
                "purple": "#00ff00",
                "red": "#00ff00",
                "white": "#00ff00",
                "yellow": "#00ff00"
            }
        ]
    }
```

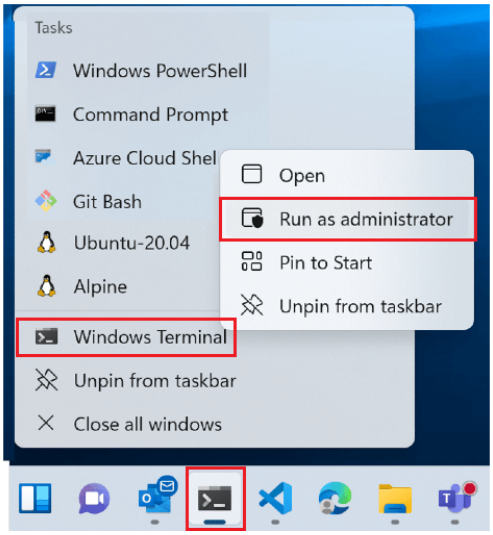
Questions fréquentes (FAQ) sur le Terminal Windows

Forum aux questions

Trouvez des réponses à certaines des questions les plus fréquemment posées sur le Terminal Windows.

Comment exécuter un shell dans le Terminal Windows en mode administrateur ?

Pour exécuter le Terminal Windows avec des autorisations administrateur élevées (mode Administrateur), cliquez avec le bouton droit sur l'icône Terminal Windows, puis à nouveau sur le titre du Terminal Windows qui s'affiche, puis sélectionnez « Exécuter en tant qu'administrateur ».



Vous pouvez aussi ouvrir le menu Accès rapide de Windows en utilisant le raccourci « Touche Windows ⊞ + X », puis sélectionner Terminal Windows (Administrateur).

Est-il possible de mélanger des onglets administrateur et non-administrateur dans une fenêtre du Terminal Windows ?

Non, mélanger des shells en onglet ayant des autorisations de niveau administrateur avec d'autres qui n'ont pas ces autorisations n'est pas pris en charge pour des raisons de sécurité.

Puis-je utiliser le Terminal Windows comme terminal intégré dans VSCode ?

Non, Visual Studio Code utilise xterm.js et est écrit en TypeScript, alors que le Terminal Windows est du code natif.

Quels sont les shells pris en charge par le Terminal Windows ?

Le Terminal Windows prend en charge toutes les lignes de commande ou tous les shells présents sur votre machine, y compris ceux qui sont inclus avec Windows, comme PowerShell ou l'invite de commandes Windows (cmd.exe) ainsi que toutes les distributions Linux qui peuvent être installées avec WSL, Azure Cloud Shell, Git Bash, etc. Le terminal détecte automatiquement l'installation d'une distribution Linux avec WSL et crée un profil pour vous. Il peut aussi enregistrer vos informations d'identification Azure, ce qui vous permet de vous connecter rapidement à Azure Cloud Shell.

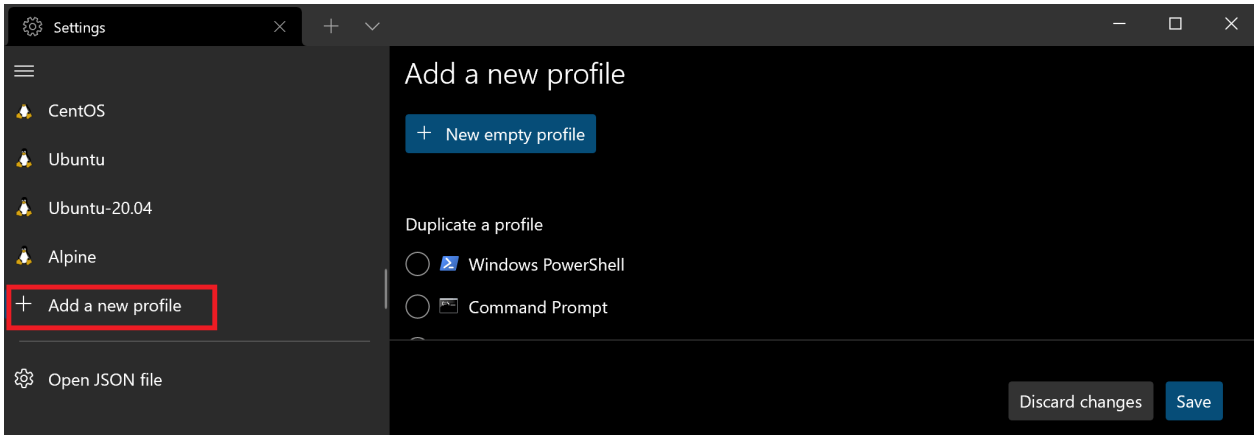
Quelle est la différence entre un interpréteur de commandes et un terminal ?

Le Terminal Windows est essentiellement un hôte qui vous permet d'exécuter plusieurs applications en ligne de commande ou interpréteurs de commandes côte à côte dans un environnement personnalisable à l'aide d'onglets ou de volets de fenêtre. Exemples d'applications « shell » : `cmd.exe` (l'invite de commandes Windows traditionnelle), `powershell` ou `zsh`. Il s'agit d'applications texte uniquement qui fournissent des flux de caractères et qui ne se soucient pas de la façon dont le rendu est effectué pour l'utilisateur. Elles sont également parfois appelées applications « clientes en ligne de commande ». En revanche, les applications de « terminal », comme le Terminal Windows, gnome-terminal, xterm, iterm2 ou hyper, sont toutes des applications graphiques qui peuvent être utilisées pour effectuer le rendu de la sortie des clients en ligne de commande, en personnalisant des éléments tels que la police, la taille du texte,

les couleurs, etc. Sur Windows, si vous exécutez `cmd.exe`, le système d'exploitation crée une instance de `conhost.exe` en tant que « terminal » pour afficher le client en ligne de commande `cmd.exe`. La même chose se produit pour PowerShell, le système crée une nouvelle fenêtre conhost pour n'importe quel client qui n'est pas déjà connecté à un terminal d'un certain type. N'importe quel terminal peut exécuter n'importe quelle application cliente en ligne de commande, de sorte que le Terminal Windows peut exécuter l'interpréteur de commandes de votre choix, comme Bash à l'aide du Sous-système Windows pour Linux (WSL).

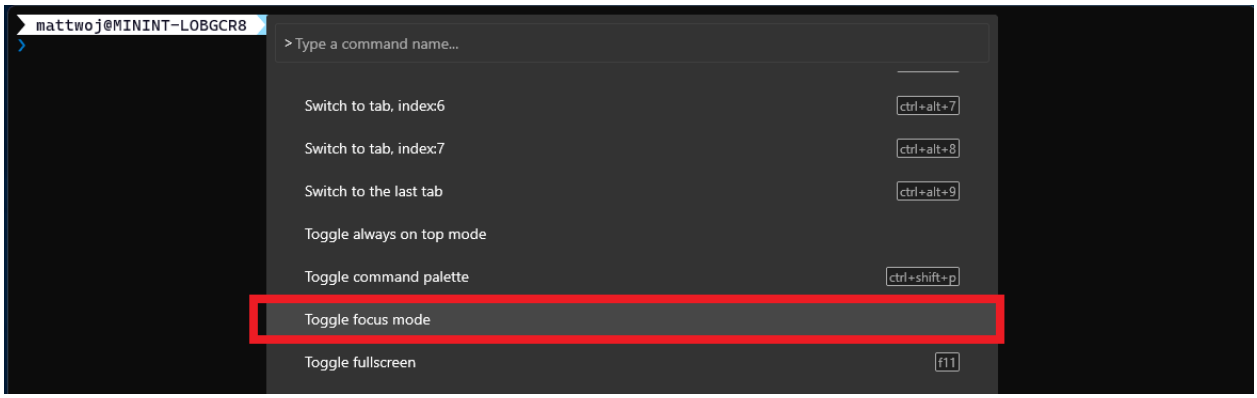
Comment puis-je ajouter manuellement un shell ?

Dans votre fichier settings.json, vous pouvez créer ou modifier des profils qui exécutent n'importe quel exécutable de ligne de commande. Dans le fichier settings.json, définissez « commandline » sur ce que vous voulez. Par exemple, powershell --> "pwsh.exe". Vous pouvez aussi ajouter un profil en utilisant l'interface utilisateur des paramètres du terminal en faisant défiler la liste des profils vers le bas et en sélectionnant « + Ajouter un nouveau profil ».

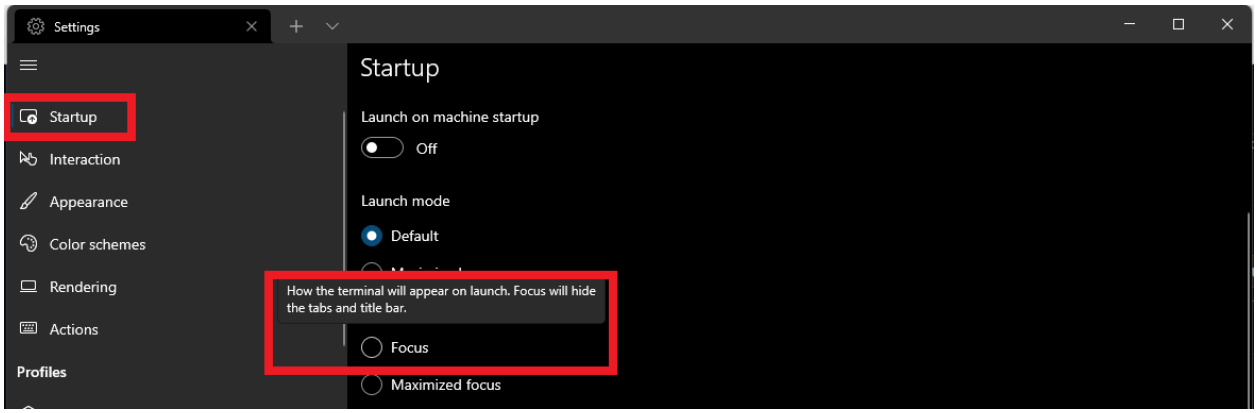


Comment quitter le mode Focus ?

Pour quitter le mode Focus, qui masque les onglets et la barre de titre dans Terminal Windows, ouvrez la palette de commandes du terminal (Ctrl+Maj+P), recherchez « activer/désactiver le mode focus », puis appuyez sur Entrée.



Vous pouvez également indiquer un mode de lancement différent de « Focus » dans les paramètres de démarrage.



Quelle est la différence entre le Terminal Windows et le Terminal Windows (Preview) ?

Terminal Windows est la version publique stable et reçoit des mises à jour régulières qui ont été testées et déboguées dans la préversion. La méthode recommandée pour l'installer est via le Microsoft Store, qui va fournir des mises à jour automatiques chaque fois qu'elles sont publiées. Terminal Windows Preview est une version destinée aux personnes qui veulent essayer les fonctionnalités les plus récentes au fur et à mesure de leur développement, tester s'il y a des bogues et faire en sorte qu'elles deviennent suffisamment stables pour être ajoutées à la version principale du Terminal. Les fonctionnalités de cette version sont documentées avec une étiquette (Préversion).

Quelles sont les autres façons d'installer le Terminal Windows ?

Nous recommandons d'installer le Terminal Windows en utilisant le Microsoft Store, mais vous pouvez aussi l'installer en utilisant le Gestionnaire de package Windows,

GitHub, Chocolatey ou Scoop.

Est-il possible d'initialiser un profil Terminal Windows avec un fichier de commandes ?

Oui. Vous devez d'abord accéder à la section Profiles de votre fichier settings.json. En utilisant la propriété "commandline":, vous pouvez spécifier un fichier de commandes, une commande, une connexion SSH ou un exécutable que vous voulez exécuter comme profil dans le Terminal Windows. Vous devez simplement entrer le chemin du fichier que vous voulez exécuter.

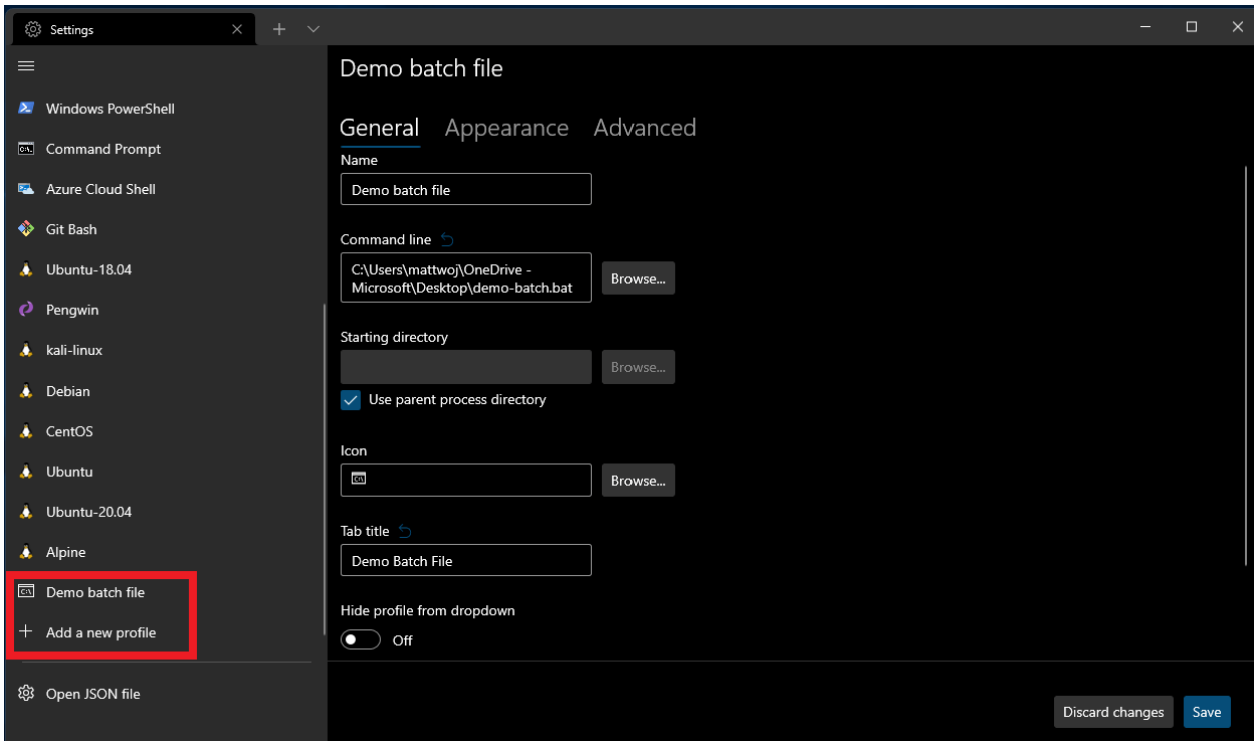
Cet exemple montre un exemple de profil de terminal configuré à partir d'un fichier de commandes « demo ».

JSON

```
{
"commandline": "%USERPROFILE%/OneDrive/demo.bat",
"name": "Batch Profile"
}
```

Vous pouvez aussi faire cela dans l'interface utilisateur Paramètres. Sélectionnez « + Ajouter un nouveau profil » > + « Nouveau profil vide ». Accédez au répertoire de démarrage où se trouve votre fichier de commandes (ou votre connexion SSH, votre exécutable, votre fichier de commandes, etc.). Donnez un nom au profil et enregistrez-

le.



De quelle sorte sont les fonctionnalités des contributeurs de la communauté open source ajoutées au Terminal Windows ?

Il y a eu un large éventail de contributions au Terminal Windows, y compris des correctifs de bogues, l'identification et la discussion de problèmes, la contribution à cette documentation, mais quelques-unes de nos fonctionnalités préférées qui proviennent des contributions de la communauté ont permis la prise en charge des images et gifs d'arrière-plan, des effets « rétro » et de la coloration des onglets, pour n'en citer que quelques-unes. En savoir plus sur la façon de contribuer.

Qu'est-ce que conhost.exe ?

L'hôte de console Windows, conhost.exe, est l'expérience utilisateur de ligne de commande d'origine de Windows. Il héberge aussi l'infrastructure de ligne de commande de Windows et le serveur d'API de la Console Windows, le moteur d'entrée, le moteur de rendu, les préférences utilisateur, etc. Un des objectifs principaux de la Console Windows est de maintenir la compatibilité descendante, ce qui signifie que l'ajout de nouvelles fonctionnalités est devenu prohibitif et a conduit à la création du

Terminal Windows. Pour plus d'informations, consultez le dépôt open source du Terminal Windows et la documentation de la Console Windows.

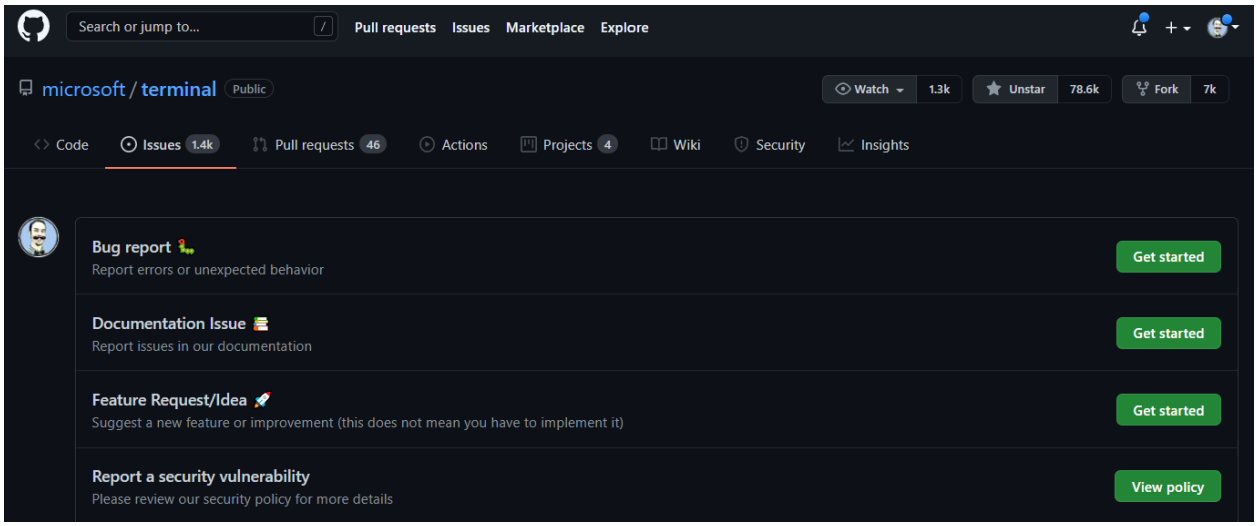
Puis-je enregistrer la disposition de toutes mes fenêtres de console ouvertes lors de la fermeture du Terminal Windows et les restaurer lors du démarrage d'une nouvelle session ?

Oui. À compter de Terminal Windows Preview v1.12.2922.0, vous pouvez enregistrer les dispositions des volets de la fenêtre lors de la fermeture d'une session de terminal avec le paramètre global firstWindowPreference.

Qu'est-ce qui est planifié pour le Terminal Windows ? Existe-t-il une feuille de route des développements ou une liste des demandes de fonctionnalités auxquelles je peux contribuer ?

Le Terminal Windows fait l'objet de développements importants. Vous pouvez consulter les plans de l'équipe dans le document Feuille de route de Terminal 2.0 dans le dépôt open source du terminal. Les nouvelles fonctionnalités vont d'abord dans Terminal Windows Preview puis, en général un mois après avoir été dans la préversion, ces fonctionnalités sont déplacées dans Terminal Windows.

Vous pouvez contribuer avec des demandes de fonctionnalités ou des idées ainsi qu'en signalant des bogues, des failles de sécurité ou des problèmes de documentation en ouvrant un problème dans le dépôt du Terminal.



Comment personnaliser les couleurs d'invite PowerShell à l'aide de PSReadLine ?

Le module PSReadLine est chargé de définir les couleurs et le comportement de votre ligne de commande PowerShell. Vous pouvez définir des couleurs pour des attributs spécifiques de votre invite PowerShell en utilisant Set-PSReadLineOption. Consultez Exemple 4 : Définir plusieurs options de couleur.

Est-il possible d'exécuter des onglets en tant qu'administrateur (élévation de privilèges) dans des fenêtres de terminal sans élévation de privilèges ?

Non. Pour plus d'informations techniques, reportez-vous à cette spécification et à ce document.

Pourquoi le thème d'apparence par défaut du Terminal Windows est-il défini sur Sombre alors que les paramètres de mon système

d’exploitation Windows sont définis sur Clair ?

Le thème système dans Windows 11 utilise par défaut l’apparence du thème Clair, sauf en cas de modification par l’utilisateur. Cependant, le modèle de couleurs qui définit l’apparence du Terminal Windows est Sombre par défaut. De nombreux utilisateurs ne modifient pas les thèmes d’apparence et ne voient que les thèmes par défaut. Si vous avez pour objectif d’aligner l’apparence du Terminal Windows avec les couleurs de la barre de titre du système d’exploitation Windows, les options suivantes s’offrent à vous : 1) Ne rien faire (valeur par défaut du Terminal avant la version 1.16). Dans la configuration par défaut, le résultat est un contraste peu attrayant sur le plan visuel entre le contenu du terminal noir et la barre de titre claire. 2) Configurer le modèle de couleurs du terminal de sorte qu’il utilise le thème du système d’exploitation et laisser le thème de l’application défini sur « système ». Dans la configuration par défaut, le terminal affiche le texte en noir sur un arrière-plan blanc. 3) Définir le thème par défaut du Terminal Windows sur Sombre, quel que soit le thème du système d’exploitation. Dans la configuration par défaut, le terminal affiche le texte en blanc sur un arrière-plan noir, avec une barre de titre sombre. L’option 3 offre le meilleur compromis pour les utilisateurs, à savoir des paramètres par défaut sans grande surprise pour les utilisateurs et une esthétique plaisante. Terminal Windows version 1.16 introduit également de nouveaux boutons bascules pour personnaliser l’apparence de la fenêtre. Vous pouvez ainsi personnaliser la couleur de la barre de titre et la couleur de l’onglet (y compris en reproduisant automatiquement l’arrière-plan), et choisir différentes couleurs pour les fenêtres avec ou sans focus. Terminal Windows version 1.17 offre plus de flexibilité puisqu’il permet de synchroniser le thème du terminal avec celui du système d’exploitation et de synchroniser le modèle de couleurs avec le thème du système d’exploitation. Le terminal est déjà composé à 99% de texte blanc sur un arrière-plan noir. Ces modifications ne font qu’aligner la barre de titre avec cette configuration.

Comment faire pour redéfinir le thème du Terminal Windows sur « système » ?

Vous pouvez ajouter `"theme": "system"` à votre `settings.json` ou modifier le thème dans la page « Apparence » de Paramètres du Terminal Windows.

Pourquoi l’onglet est-il toujours noir quand je choisis le thème Clair pour le

Terminal Windows ?

Il s'agit d'un effet secondaire des modifications apportées au thème dans la version 1.16. Les thèmes par défaut dans les versions 1.16 et ultérieures utilisent toujours la couleur d'arrière-plan du terminal comme couleur par défaut pour chaque onglet. Par défaut, dans une fenêtre de terminal noire, vous obtenez un onglet noir. Avec un modèle de couleurs bleu (comme Campbell PowerShell), vous obtenez un onglet bleu. Il s'agit de donner au terminal une sensation de « transparence ». En mode Clair, vous vous retrouvez avec un onglet noir sur une rangée d'onglets blancs. Toutefois, avec les versions 1.16 et ultérieures, vous pouvez personnaliser le thème du terminal. Par exemple :

JSON

```
"theme": "White Tabs",
"themes":
[
    {
        "name": "White Tabs",
        "tab":
        {
            "background": "#ffffffff",
        },
        "window":
        {
            "applicationTheme": "light"
        }
    },
]
```

Si vous utilisez un thème Clair pour le système d'exploitation et que vous souhaitez définir le modèle de couleurs du terminal avec un arrière-plan blanc, la version 1.17 vous permet de définir différents modèles de couleurs en fonction du thème de la fenêtre. Par exemple, pour modifier la couleur d'arrière-plan en fonction du `window.applicationTheme` de Terminal Windows, vous pouvez effectuer les opérations suivantes :

JSON

```
"colorScheme":
{
    "light": "One Half Light",
    "dark": "One Half Dark",
},
```

Collaborer avec nous sur GitHub

La source de ce contenu se trouve sur GitHub, où vous pouvez également créer et examiner les problèmes et les demandes de tirage. Pour plus d'informations, consultez notre guide du contributeur.

Commentaires sur Windows Terminal

Windows Terminal est un projet open source. Sélectionnez un lien pour fournir des commentaires :

Ouvrir un problème de documentation

Indiquer des commentaires sur le produit

Exemple de code Terminal Windows

Article • 21/03/2023

Explorez quelques exemples de code hébergés dans le dépôt Terminal Windows, y compris des exemples de nuanceur de pixels .hlsl, un exemple de pseudoconsole Win32 EchoCon ConPTY, un exemple de console WPF GUIConsole ciblant .NET, un exemple MiniTerm utilisant des appels d'API PTY de base et une démonstration ReadConsoleInputStream pour la supervision des événements de console lors du streaming d'entrées de caractères.

Nuanceurs de pixels

En raison de la grande puissance de calcul des GPU, vous pouvez faire des choses étonnantes avec des nuanceurs de pixels, comme le zoom fractal en temps réel, les traceurs de rayon et le traitement d'image.

Le Terminal Windows permet aux utilisateurs de fournir un nuanceur de pixels, qui est appliqué au terminal en ajoutant la propriété `experimental.pixelShaderPath` à un profil dans votre fichier settings.json. Les nuanceurs de pixels sont écrits dans un langage appelé HLSL, qui est un langage de type C avec quelques restrictions.

Essayez un des nombreux exemples de nuanceur de pixels .hlsl fournis dans dépôt du Terminal Windows : Nuanceurs de pixels.

Exemple d'application EchoCon ConPTY

Cet exemple d'application montre comment utiliser la pseudoconsole Win32 (ConPTY) :

1. En créant un canal d'entrée et un canal de sortie
2. En appelant CreatePseudoConsole() pour créer une instance ConPTY attachée à l'autre extrémité des canaux
3. En générant une instance de ping.exe connectée à ConPTY
4. En exécutant un thread qui écoute la sortie de ping.exe et qui écrit le texte reçu sur la console

Visitez le dépôt du Terminal Windows pour trouver cet exemple : Exemple d'application EchoCon ConPTY.

Exemple d'application GUIConsole

Cet exemple d'application fournit un exemple de squelette d'une console WPF personnalisée.

Dans cet exemple, vous allez trouver :

- GUIConsole.WPF : une application WPF ciblant .NET 4.6.1, qui crée une seule fenêtre WPF agissant comme console et qui laisse la console sous-jacente visible.

- GUIConsole.ConPTY : une bibliothèque .NET Standard 2.0 qui gère la création de la console et active le comportement de pseudoconsole. Le fichier Terminal.cs contient les éléments visibles publiquement avec lesquels l'application WPF va interagir. Terminal.cs expose deux éléments qui permettent la lecture et l'écriture dans la console :
  - `ConsoleOutStream` : un FileStream raccordé au canal de sortie de la pseudoconsole. La sortie sera du VT100.
  - `WriteToPseudoConsole`(entrée chaîne) : une méthode qui va prendre la chaîne donnée et l'écrire dans la pseudoconsole via son canal d'entrée. Ceci accepte du VT100.

Visitez le dépôt du Terminal Windows pour trouver cet exemple : GUIConsole.

Exemple d'application MiniTerm

Ce terminal expérimental montre des appels d'API de base (non destinés à une utilisation réelle) en utilisant les API PTY de Microsoft. Écrit en C# et largement basé sur les exemples de code natif.

Visitez le dépôt du Terminal Windows pour trouver cet exemple : MiniTerm.

Démonstration ReadConsoleInputStream

Démonstration de la supervision asynchrone des événements de console (comme des événements de souris, de menu, de focus, de redimensionnement de la mémoire tampon/fenêtre d'affichage) tout en effectuant simultanément le streaming de la vue de saisie de caractères à partir de la console. Une fonctionnalité particulièrement utile quand vous travaillez avec des flux VT100 et ConPTY.

Visitez le dépôt du Terminal Windows pour trouver cette démonstration : Démonstration ReadConsoleInputStream.

Résolution des problèmes dans le Terminal Windows

Article • 21/03/2023

Ce guide aborde des erreurs et obstacles courants que vous pouvez rencontrer lors de l'utilisation du Terminal Windows.

L'ouverture des paramètres ne donne rien (ou ouvre une application inattendue)

Si vous cliquez sur le bouton Paramètres dans la liste déroulante, le terminal tente d'ouvrir le fichier de paramètres `settings.json`. Le système d'exploitation tente alors de lancer votre éditeur de fichiers `.json` configuré. Il peut s'agir de Visual Studio, du Bloc-notes ou d'une autre application complètement inattendue. Si aucun éditeur `.json` n'est configuré sur votre ordinateur, le système d'exploitation affiche la boîte de dialogue « Comment voulez-vous ouvrir ce fichier ? ».

> **Conseil**
>
> Vous pouvez également utiliser l'interface utilisateur des paramètres pour configurer vos paramètres si vous utilisez la **préversion du Terminal Windows Preview**. Découvrez comment ouvrir l'interface utilisateur des paramètres dans la **page Actions**.

Configurez votre distribution WSL pour qu'elle démarre dans le répertoire de base `~` lorsqu'elle est lancée dans les anciennes versions de Windows Terminal

Par défaut, le paramètre `startingDirectory` d'un profil est `%USERPROFILE%` (`C:\Users\<YourUsername>`). Il s'agit d'un chemin d'accès Windows. Pour les distributions WSL exécutant une nouvelle version de Windows Terminal, les systèmes de fichiers peuvent entrer `~` pour définir ce chemin d'accès. Dans les anciennes versions de Windows Terminal, vous pouvez utiliser `/home/<Your Ubuntu Username>` pour vous référer

directement à votre dossier de départ. Par exemple, le paramètre suivant lance la distribution « Ubuntu-20.04 » dans son chemin de fichier d'accueil :

JSON

```
{
    "name": "Ubuntu-20.04",
    "commandline" : "wsl -d Ubuntu-20.04",
    "startingDirectory" : "/home/<Your Ubuntu Username>"
}
```

Si vous utilisez une version très ancienne de Windows Terminal, WSL peut nécessiter l'utilisation du préfixe `\\wsl$\` lors de la référence au chemin d'accès d'accueil d'une distribution pour le paramètre `startingDirectory`. Par exemple, le paramètre suivant lance la distribution « Ubuntu-18.04 » dans son chemin d'accès de base :

JSON

```
{
    "name": "Ubuntu-18.04",
    "commandline" : "wsl -d Ubuntu-18.04",
    "startingDirectory" : "//wsl$/Ubuntu-18.04/home/<Your Ubuntu Username>"
}
```

> ⓘ **Important**
>
> Sur les versions plus récentes de Windows, `startingDirectory` peut accepter des chemins de type Linux.

Définition du titre de l'onglet

Pour que le shell définisse automatiquement le titre de votre onglet, visitez le didacticiel Définir le titre des onglets. Si vous souhaitez définir votre propre titre d'onglet, ouvrez le fichier settings.json et effectuez les étapes suivantes :

1. Dans le profil pour la ligne de commande de votre choix, ajoutez `"suppressApplicationTitle": true` pour supprimer tous les événements de modification de titre qui sont envoyés à partir de l'interpréteur de commandes. Ajouter *uniquement* ce paramètre à votre profil définit le titre de l'onglet sur le nom de votre profil.

2. Si vous souhaitez un titre d'onglet personnalisé qui n'est pas le nom de votre profil, ajoutez `"tabTitle": "TITLE"`. Remplacez « TITLE » par le titre d'onglet de votre choix.

Arguments de ligne de commande dans PowerShell

Visitez la page Arguments de ligne de commande pour découvrir comment les arguments de ligne de commande fonctionnent dans PowerShell.

Arguments de ligne de commande dans WSL

Visitez la page Arguments de ligne de commande pour découvrir comment les arguments de ligne de commande fonctionnent dans WSL.

Problème de réglage de `startingDirectory`

Si `startingDirectory` est ignoré dans votre profil, commencez par vérifier que la syntaxe du fichier settings.json est correcte. Pour vous aider à vérifier cette syntaxe, `"$schema": "https://aka.ms/terminal-profiles-schema"` est automatiquement injecté. Certaines applications, comme Visual Studio Code, peuvent utiliser ce schéma injecté pour valider votre fichier json au fur et à mesure que vous lui apportez des modifications.

Si vos paramètres sont corrects, vous pouvez exécuter un script de démarrage qui définit le répertoire de démarrage de votre terminal séparément. Par exemple, PowerShell possède son propre concept distinct de profils. Si vous modifiez votre répertoire de départ, celui-ci aura la priorité sur le paramètre défini dans le Terminal Windows.

Si vous exécutez un script à l'aide du paramètre de profil `commandline`, vous pouvez également définir l'emplacement ici. À l'instar des profils PowerShell, vos commandes ont la priorité sur le paramètre de profil `startingDirectory`.

L'objectif de `startingDirectory` est de lancer une nouvelle instance du Terminal Windows dans le répertoire donné. Si le terminal exécute du code qui modifie son répertoire, il peut être intéressant d'y jeter un coup d'œil.

Ctrl + = n'augmente pas la taille de police

Si vous utilisez une disposition de clavier allemande, vous risquez de rencontrer ce problème. `Ctrl + =` est désérialisé avec `Ctrl + Maj + 0` si la disposition de clavier principale est définie sur l'allemand. Il s'agit du mappage approprié pour les claviers allemands.

Surtout, l'application ne reçoit jamais la séquence de touches `Ctrl + Maj + 0`. Cela est dû au fait que `Ctrl + Maj + 0` est réservé par Windows si vous avez plusieurs dispositions de clavier actives.

Si vous souhaitez désactiver cette fonctionnalité afin que `Ctrl+=` fonctionne correctement, suivez les instructions pour « Modifier les raccourcis pour basculer la disposition du clavier dans Windows 10 » dans ce billet de blog.

Modifiez l'option « Changer de disposition du clavier » sur « Non attribué » (ou désactivez `Ctrl + Maj`), puis sélectionnez **OK** puis **Appliquer**. `Ctrl + Maj + 0` doit maintenant fonctionner comme une combinaison de touches et est passée au terminal.

En revanche, si vous utilisez cette fonctionnalité de raccourci clavier pour plusieurs langues d'entrée, vous pouvez configurer votre propre combinaison de touches personnalisée dans votre fichier settings.json.

Le texte est flou

Certains pilotes d'affichage et combinaisons matérielles ne gèrent pas les zones de défilement et/ou modifiées sans brouiller les données de l'image précédente. Pour atténuer ce problème, vous pouvez ajouter une combinaison de ces paramètres de rendu globaux afin de réduire la tension imposée à votre matériel par le renderer de texte de terminal.

Mes couleurs semblent bizarres. L'écran contient des barres noires.

> **Important**
>
> Cela s'applique uniquement à la version 1.2+ du Terminal Windows. Si vous rencontrez des problèmes de couleur dans le Terminal Windows 1.0 ou 1.1 ou des problèmes qui ne sont pas capturés ici, signalez un bogue.

Le Terminal Windows 1.2 et ultérieur a une meilleure compréhension de certains paramètres de couleur d'application. Du fait de cette meilleure compréhension, nous

avons pu supprimer un certain nombre de blocs de compatibilité qui entraînaient une mauvaise expérience utilisateur. Malheureusement, des problèmes peuvent se poser pour quelques applications .

Cette page de dépannage sera mise à jour avec la liste des problèmes connus et leurs solutions de contournement.

Lignes noires dans PowerShell (5.1, 6.x, 7.0)

Couplé à la bibliothèque d'édition de ligne PSReadline de PowerShell, le Terminal peut dessiner des lignes noires à l'écran. Ces zones mal colorées s'étendent à l'écran au-delà de votre invite de commandes, partout où vous avez des paramètres de commande, des chaînes ou des opérateurs.

La version **2.0.3** de PSReadline a été publiée et contient un correctif pour résoudre ce problème. Si vous utilisez la préversion de PSReadline, notez qu'aucun correctif n'est encore disponible.

Pour effectuer la mise à jour vers la dernière version de PSReadline, exécutez la commande suivante :

PowerShell

```
Update-Module PSReadline
```

Pourquoi mes emojis n'apparaissent-ils pas sous forme d'icônes dans la liste de raccourcis ?

Seules les images liées à partir d'un emplacement de fichier peuvent être affichées sous forme d'icônes de profil dans la liste de raccourcis. Les emojis ne sont pas pris en charge comme icônes de liste de raccourcis.

Notes techniques

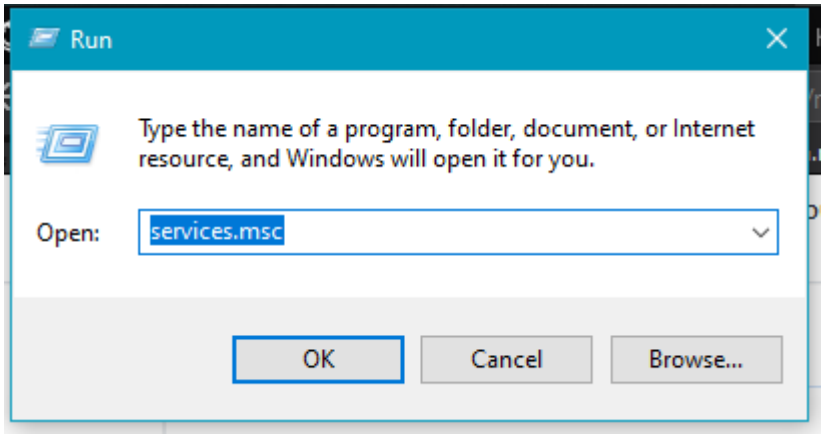
Les applications qui utilisent la famille d'API `GetConsoleScreenBufferInfo` pour récupérer les couleurs actives de la console au format Win32, puis tentent de les transformer en séquences VT interplateformes (par exemple, en transformant `BACKGROUND_RED` en `\x1b[41m`), peuvent interférer avec la capacité du terminal à détecter la couleur d'arrière-plan que l'application tente d'utiliser.

Les développeurs d'applications sont encouragés à choisir des fonctions d'API Windows *ou* des séquences VT pour ajuster les couleurs sans les mélanger.

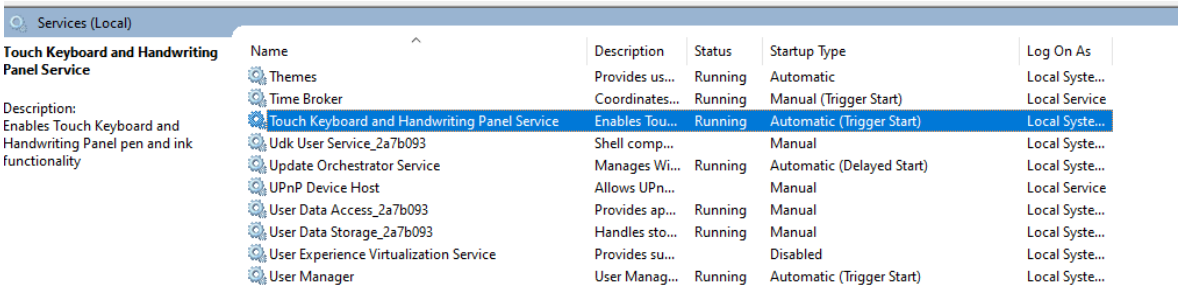
Avertissement du service du clavier

À compter du Terminal Windows 1.5, le Terminal affiche un avertissement si le « Service du clavier tactile et du volet d'écriture manuscrite » est désactivé. Ce service est nécessaire au système d'exploitation pour acheminer correctement les événements d'entrée à l'application Terminal (ainsi qu'à bien d'autres applications sur Windows). Si vous voyez cet avertissement, vous pouvez effectuer ces étapes pour réactiver le service :

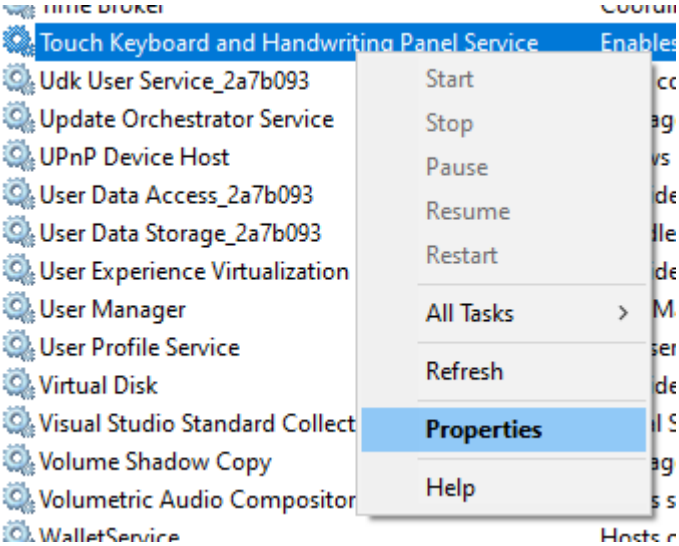
1. Dans la boîte de dialogue Exécuter, exécutez `services.msc`.



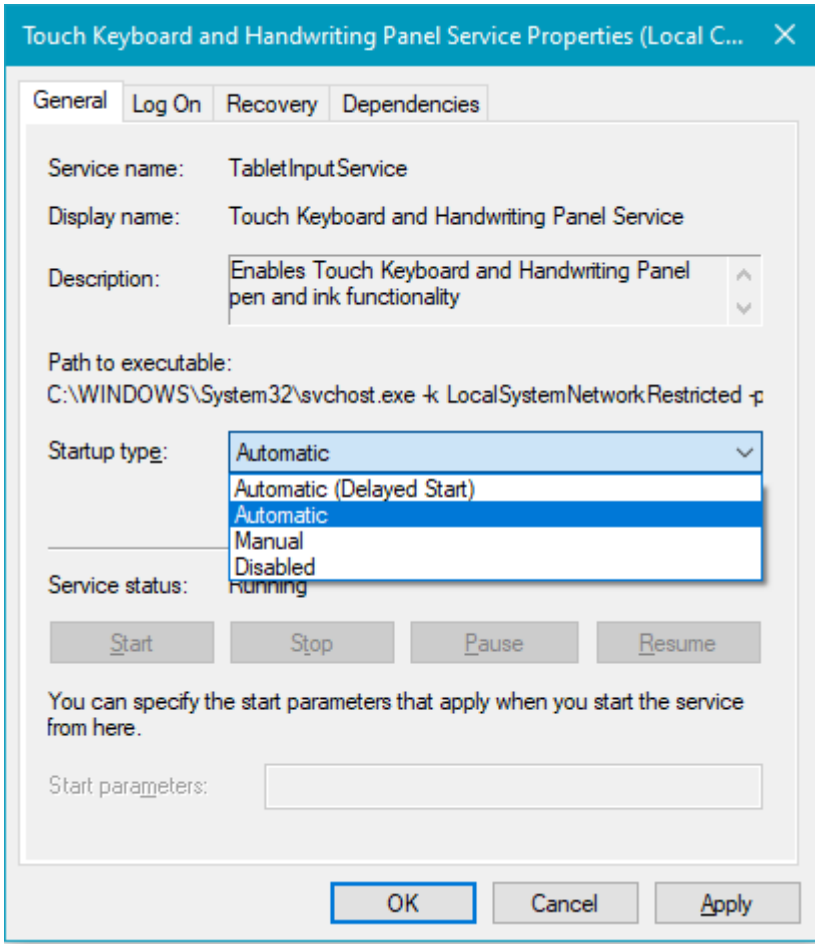
2. Recherchez « Service du clavier tactile et du volet d'écriture manuscrite ».



3. Ouvrir les « Propriétés » pour ce service.



4. Remplacez le « Type de démarrage » par « Automatique ».



5. Appuyez sur OK, puis redémarrez le PC.

Après le redémarrage de l'ordinateur, le service doit démarrer automatiquement et la boîte de dialogue ne doit plus apparaître.

Pourquoi un clignotement se produit-il quand j'utilise une ligne de commande git bash ?

Vous pouvez constater un clignotement lors de l'utilisation d'une ligne de commande git bash dans le Terminal Windows. Ce comportement est tout à fait normal. Le terminal obéit à ce que git bash lui dit de faire (définition de bell-style comme visible, ce qui entraîne l'association d'un clignotement à la réponse de bell), mais nous comprenons que cela peut être gênant. Pour résoudre ce problème, ouvrez le fichier `.inputrc` de git bash avec un éditeur de texte. Vous trouverez probablement ce fichier dans le chemin `C:\Program Files\Git\etc`. Pour l'ouvrir avec l'éditeur de texte Nano : `nano ~/.inputrc`

Modifiez la commande par défaut :

```Bash
# none, visible or audible
set bell-style visible
```

Définissez bell-style avec `none` ou `audible` pour supprimer le clignotement visible :

```Bash
set bell-style none
```

Appuyez sur Ctrl + O et Ctrl + X pour enregistrer et quitter.

Comment rétablir les paramètres par défaut dans le Terminal Windows ?

Pour rétablir les paramètres par défaut d'origine, supprimez votre fichier settings.json. Le Terminal Windows regénère alors un fichier settings.json avec les paramètres par défaut d'origine.

> **ⓘ Important**
>
> À partir du Terminal Windows version 1.10 ou ultérieure, vous devez également supprimer le fichier `state.json` situé dans le même répertoire que le fichier `settings.json` pour rétablir complètement les paramètres par défaut.

Pourquoi l'opacité acrylique ne rend-elle pas l'arrière-plan de mon terminal Windows

transparent ?

Vous pouvez définir la transparence d'une fenêtre de terminal avec la useAcrylic propriété. Il existe plusieurs raisons pour lesquelles votre paramètre d'opacité peut ne pas fonctionner pour l'acrylique, notamment :

- En tant que stratégie à l'échelle du système, l'acrylique n'est activé que pour la fenêtre de premier plan. Donc, si vous activez une autre fenêtre que le Terminal, l'acrylique du Terminal s'éteindra.
- L'acrylique ne fonctionne pas si votre matériel GPU ne le prend pas en charge. Si vous exécutez une application sur une machine virtuelle (VM) ou sur un poste de travail distant, l'acrylique ne fonctionnera probablement pas.
- L'acrylique peut être désactivé par le système d'exploitation pour un certain nombre de raisons, comme être en mode d'économie d'énergie (batterie faible) ou lors de l'accès à une machine à l'aide de Remote Desktop.

Pourquoi le pointeur de ma souris disparaît-il lorsque je survole une fenêtre et que je tape ?

Ce comportement de masquage automatique du curseur est de par sa conception, mais peut être désactivé en recherchant dans les paramètres Windows "Paramètres de la souris" > "Paramètres supplémentaires de la souris" > "Propriétés de la souris" > "Options du pointeur" > Décochez "Masquer le pointeur pendant la frappe". Vous devrez peut-être redémarrer votre terminal Windows pour que cette modification prenne effet.