Package: tutoquartotypst (via r-universe)

June 10, 2026

Title Paquet Compagnon du Tutoriel Quarto + Typst (Rencontres R 2026)

Version 0.0.0.9001

Description Fonctions utilitaires pour preparer et suivre le tutoriel
``PDF sans frictions : Typst dans vos projets Quarto''
(Rencontres R 2026). Le paquet verifie l'environnement (R,
Quarto, Typst, paquets), installe localement les exercices et
aide au rendu Typst. Installer le paquet tire automatiquement
tous les prerequis du tutoriel.

License MIT + file LICENSE

URL https://cderv.r-universe.dev/tutoquartotypst,
https://cderv.github.io/tuto-quarto-typst-rr-2026,
https://github.com/cderv/tuto-quarto-typst-rr-2026

BugReports https://github.com/cderv/tuto-quarto-typst-rr-2026/issues

Depends R (>= 4.1)

Imports brand.yml, cli, dplyr, ggplot2, ggrepel, gt, prismatic,
quarto, rlang, scales, withr, xfun, yaml

Suggests pkgdown, rstudioapi, testthat (>= 3.0.0)

SystemRequirements Quarto (>= 1.9.0), Typst (bundled with Quarto)

Config/testthat/edition 3

Config/roxygen2/markdown TRUE

Config/roxygen2/version 8.0.0

X-schema.org-keywords quarto, typst, reproducible-research, tutorial,
pdf

Encoding UTF-8

Language fr

Config/pak/sysreqs cmake make libuv1-dev libxml2-dev libssl-dev libnode-dev

Repository https://cderv.r-universe.dev

Date/Publication 2026-06-10 19:46:45 UTC

RemoteUrl https://github.com/cderv/tuto-quarto-typst-rr-2026

RemoteRef HEAD

RemoteSha 21aa496e8f027d85ff88b005e59b8dd61972fc10

RemoteSubdir pkg

Contents

`appliquer_polices_locales`
Appliquer le contournement `font-paths` *(livre, Quarto < 1.10.4)*

Description

Sur Quarto antérieur à 1.10.4, un projet livre avec `_brand.yml` a besoin de l'option `font-paths` dans `_quarto.yml` pour trouver les polices. Cette fonction l'ajoute si nécessaire (et ne fait rien si votre Quarto est assez récent ou si l'option est déjà présente). Le `_quarto.yml` est sauvegardé.

Usage

```
appliquer_polices_locales(projet = ".")
```

Arguments

`projet`	Dossier du projet livre (contenant `_quarto.yml`). Par défaut le répertoire courant.

Value

Invisiblement, `TRUE` si le fichier a été modifié, `FALSE` sinon.

Examples

```
appliquer_polices_locales()
```

`basculer_charte` *Appliquer une variante de charte Star Wars*

Description

Dépose l'une des chartes thématiques (`empire`, `jedi`, `mando`) comme `_brand.yml` du projet, pour illustrer qu'**un même projet change d'identité en changeant simplement de charte**. À utiliser **après** l'exercice. Votre `_brand.yml` est sauvegardé avant remplacement.

Usage

```
basculer_charte(variante = c("empire", "jedi", "mando"), projet = ".")
```

Arguments

`variante`	`"empire"` (défaut), `"jedi"` ou `"mando"`.
`projet`	Dossier du projet. Par défaut le répertoire courant.

Value

Invisiblement, le chemin du `_brand.yml`.

Examples

```
basculer_charte("jedi")
```

basculer_hors_ligne *Basculer un exercice en mode hors-ligne (polices Inter locales)*

Description

Quand le réseau manque, Inter (déclarée `source: google` dans `_brand.yml`) ne peut pas être téléchargée. Cette fonction dépose les fichiers Inter embarqués dans `_fonts/` et bascule l'entrée Inter de `_brand.yml` en `source: file` — **sans toucher** au reste de votre charte (couleurs, etc.). Votre `_brand.yml` est sauvegardé avant modification.

Usage

```
basculer_hors_ligne(projet = ".", retour = FALSE)
```

Arguments

`projet`	Dossier de l'exercice (contenant `_brand.yml`). Par défaut le répertoire courant.
`retour`	Logique. Restaurer le `_brand.yml` d'origine (revenir en ligne) ? Par défaut `FALSE`.

Value

Invisiblement, le chemin du `_brand.yml`, ou `NULL` si rien n'a été modifié.

Examples

```
basculer_hors_ligne()
basculer_hors_ligne(retour = TRUE)
```

comparer_chartes *Comparer les couleurs des variantes de charte*

Description

Affiche la couleur principale (`primary`) de chaque variante de charte, avec un aperçu coloré (si votre console le permet).

Usage

```
comparer_chartes()
```

Value

Invisiblement, un vecteur nommé des couleurs principales.

Examples

```
comparer_chartes()
```

`creer_projet_typst` *Créer un projet Quarto + Typst réutilisable*

Description

Génère un squelette de projet Typst prêt à l'emploi (hors thème Star Wars), pour réutiliser ce que vous avez appris **après** le tutoriel : un document ou un livre, avec une charte `_brand.yml` à adapter. Sur Quarto < 1.10.4, le contournement `font-paths` est ajouté automatiquement aux projets livre.

Usage

```
creer_projet_typst(
  dest,
  type = c("document", "livre"),
  brand = TRUE,
  offline = FALSE
)
```

Arguments

`dest`	Dossier à créer pour le projet.
`type`	`"document"` (défaut) pour un `.qmd` unique, ou `"livre"` pour un projet livre multi-chapitres.
`brand`	Logique. Inclure une charte `_brand.yml` à adapter ? Défaut `TRUE`.
`offline`	Logique. Inclure les polices Inter en local (`source: file`) ? Défaut `FALSE`.

Value

Invisiblement, le chemin absolu du projet créé.

Examples

```
creer_projet_typst("mon-rapport")
creer_projet_typst("mon-livre", type = "livre")
```

diagnostic_typst *Diagnostic détaillé de la chaîne Quarto / Typst*

Description

Agrège les informations utiles au dépannage : chemin et version de Quarto, version de Typst embarqué, présence du cache de polices, et verdict sur le besoin du contournement `font-paths` (Quarto < 1.10.4).

Usage

```
diagnostic_typst(projet = ".")
```

Arguments

`projet`	Dossier de projet où chercher le cache de polices `.quarto/typst/fonts`. Par défaut le répertoire courant.

Value

Invisiblement, une liste des informations collectées.

Examples

```
diagnostic_typst()
```

diagnostiquer_rendu *Décoder une erreur de rendu Quarto / Typst*

Description

Traduit en français les messages d'erreur ou d'avertissement les plus fréquents lors du rendu Typst, et indique s'ils sont **bénins** (le PDF est quand même produit) ou **bloquants**, avec l'action à mener.

Usage

```
diagnostiquer_rendu(texte = NULL)
```

Arguments

`texte`	Le texte de l'erreur (copié depuis la console). Si `NULL` (défaut), affiche la liste de référence des cas connus.

Value

Invisiblement, un vecteur des gravités reconnues (`"benin"` / `"bloquant"`), ou `NULL` si `texte` est `NULL`. Appelée surtout pour son affichage.

Examples

```
diagnostiquer_rendu()
diagnostiquer_rendu("Error: unknown font family 'Inter'")
```

---

exporter_diagnostic *Exporter un diagnostic d'installation dans un fichier*

Description

Affiche un résumé de votre environnement (R, Quarto, paquets) dans la console, à copier-coller si vous demandez de l'aide. Peut aussi l'écrire dans un fichier.

Usage

```
exporter_diagnostic(fichier = NULL)
```

Arguments

`fichier`	Chemin d'un fichier où écrire **aussi** le diagnostic. Par défaut `NULL` : affichage console uniquement (rien à ouvrir).

Value

Invisiblement, les lignes du diagnostic.

Examples

```
exporter_diagnostic()
```

---

inspecter_typ *Inspecter le code Typst intermédiaire d'un document*

Description

Rend un `.qmd` `format: typst` en conservant le `.typ` intermédiaire (`keep-typ`), pour comprendre la couche Typst générée par Quarto. Pensé pour un **document** simple (l'exercice 1) ; pour un livre, le comportement est moins prévisible.

Usage

```
inspecter_typ(qmd, ouvrir = TRUE)
```

Arguments

qmd	Chemin du fichier .qmd à rendre.
ouvrir	Logique. Ouvrir le .typ produit dans l'éditeur ? Par défaut TRUE.

Details

À utiliser **après** avoir produit votre propre PDF : c'est un outil pour comprendre la couche Typst, pas un raccourci vers la solution de l'exercice.

Value

Invisiblement, le chemin du .typ produit, ou NULL en cas d'échec.

Examples

```
inspecter_typ("rapport-starwars.qmd")
```

---

installer_exercices *Installer les exercices du tutoriel*

Description

Copie les fichiers de départ (« starters ») des exercices, embarqués dans le paquet, vers un dossier de travail local. Seuls les starter/ sont copiés ; les corrections ne sont pas posées ici (voir ouvrir_correction() pour les consulter en ligne, recuperer_correction() pour les copier en local).

Usage

```
installer_exercices(dest = NULL, quels = c("tous", "01", "02"), force = FALSE)
```

Arguments

dest	Chemin du dossier de destination (créé si besoin). Par défaut NULL : en session interactive, le dossier est **demandé** (sélecteur RStudio si disponible, sinon invite texte ; valeur proposée "exercices-typst"). Fournir un chemin explicite court-circuite la demande (utile en script). En mode non-interactif sans dest, "exercices-typst" (dans le répertoire courant) est utilisé.
quels	Quels exercices installer : "tous" (défaut), "01" (document Typst) ou "02" (projet livre).
force	Logique. Passer la confirmation interactive **et** écraser un dossier de destination déjà existant et non vide ? Par défaut FALSE (en mode non-interactif, force = TRUE est requis pour confirmer la copie).

Value

Invisiblement, le chemin absolu du dossier de destination, ou `NULL` si l'installation a été annulée.

Examples

```
installer_exercices()
installer_exercices(quels = "01")
```

---

lister_exercices *Lister les exercices du tutoriel*

Description

Affiche les exercices disponibles avec, pour chacun, son intention. Sert à s'orienter ; ne dévoile pas les étapes de résolution.

Usage

```
lister_exercices()
```

Value

Invisiblement, les codes des exercices (`character`).

Examples

```
lister_exercices()
```

---

nettoyer_cache *Nettoyer les artefacts de rendu d'un projet*

Description

Supprime les artefacts de rendu (`_book/`, `*_files/`, `*.typ`) pour repartir d'un état propre. Ne touche jamais à vos sources ni à `_fonts/`.

Usage

```
nettoyer_cache(projet = ".", polices = FALSE)
```

Arguments

`projet`	Dossier de projet à nettoyer. Par défaut le répertoire courant.
`polices`	Logique. Vider aussi le cache de polices `.quarto/typst/fonts` (force le re-téléchargement des polices Google au prochain rendu) ? Par défaut `FALSE`.

Value

Invisiblement, le nombre d'éléments supprimés.

Examples

```
nettoyer_cache()
```

`ouvrir_correction` *Ouvrir la correction d'un exercice (en ligne)*

Description

Ouvre, **en ligne** sur GitHub, le dossier `correction/` d'un exercice. Les corrections ne sont volontairement pas posées par `installer_exercices()` : elles sont plus utiles **après** avoir cherché par vous-même. Une confirmation est demandée. Pour en obtenir une **copie locale à retravailler**, voir `recuperer_correction()`.

Usage

```
ouvrir_correction(quel = c("01", "02"), je_confirme = FALSE)
```

Arguments

`quel`	`"01"` (défaut) ou `"02"`.
`je_confirme`	Logique. Passer la confirmation (utile en script). Défaut `FALSE`.

Value

Invisiblement, l'URL de la correction, ou `NULL` si annulé.

See Also

`recuperer_correction()` pour copier la correction en local.

Examples

```
ouvrir_correction("01")
```

ouvrir_exercices *Retrouver et ouvrir le dossier des exercices installés*

Description

Filet de sécurité quand on a perdu le message de `installer_exercices()` : ré-affiche le chemin absolu et ouvre le dossier.

Usage

```
ouvrir_exercices(dossier = "exercices-typst")
```

Arguments

`dossier`	Dossier où les exercices ont été installés. Par défaut `"exercices-typst"`.

Value

Invisiblement, le chemin absolu du dossier.

Examples

```
ouvrir_exercices()
```

par_ou_commencer *Par où commencer ?*

Description

Boussole : détecte l'état de votre préparation (paquets, Quarto, exercices installés) et indique la prochaine action à effectuer. En cas de doute, c'est la fonction à lancer.

Usage

```
par_ou_commencer(dossier = "exercices-typst")
```

Arguments

`dossier`	Dossier où chercher les exercices installés. Par défaut `"exercices-typst"`.

Value

Invisiblement, un mot-clé de l'étape courante (`character`).

Examples

```
par_ou_commencer()
```

polices_typst *Lister les polices vues par Typst*

Description

Appelle `quarto typst fonts` pour lister les familles de polices visibles par Typst, et vérifie la présence d'Inter et de Star Jedi.

Usage

```
polices_typst(projet = NULL)
```

Arguments

`projet`	Dossier de projet. Si fourni, son sous-dossier `_fonts/` est ajouté au chemin de recherche (`--font-path`).

Details

Une police déclarée `source: google` dans `_brand.yml` (Inter par défaut) n'apparaît **pas** ici tant qu'un premier rendu ne l'a pas téléchargée dans le cache. Seules les polices `source: file` (et système) sont visibles d'emblée.

Value

Invisiblement, le vecteur des familles de polices.

Examples

```
polices_typst()
```

recuperer_correction *Récupérer la correction d'un exercice en local*

Description

Copie les **sources** de la correction d'un exercice (embarquées dans le paquet) vers un dossier de travail local, pour la **retravailler** après coup. Comme `ouvrir_correction()`, une confirmation est demandée : une correction est plus utile une fois que vous avez cherché par vous-même.

Usage

```
recuperer_correction(
  quel = c("01", "02"),
  dest = "exercices-typst",
  force = FALSE
)
```

Arguments

quel	"01" (défaut) ou "02".
dest	Dossier de travail où poser la correction (créé si besoin). Par défaut "exercices-typst" — la correction atterrit alors dans exercices-typst/<exercice>/correction/, à côté du starter/.
force	Logique. Passer la confirmation **et** écraser une correction déjà copiée et non vide ? Par défaut FALSE (en mode non-interactif, force = TRUE est requis pour confirmer la copie).

Details

Par défaut, la correction est posée à côté du starter/ correspondant (dans exercices-typst/<exercice>/correction/), ce qui reproduit l'arborescence du dépôt. Aucun artefact de rendu n'est embarqué : pour obtenir le PDF / livre, rendez la correction avec quarto render.

Value

Invisiblement, le chemin du dossier de correction copié, ou NULL si annulé.

Examples

```
recuperer_correction("01")
```

---

reinitialiser_exercice

Réinitialiser un exercice à son état de départ

Description

Restaure le starter/ d'un exercice (depuis la copie embarquée dans le paquet) lorsque vous avez cassé vos fichiers. Votre dossier actuel est **sauvegardé** (jamais supprimé) avant d'être remplacé.

Usage

```
reinitialiser_exercice(
  quel = c("01", "02", "00"),
  dossier = "exercices-typst",
  force = FALSE
)
```

Arguments

quel	Quel exercice réinitialiser : "01" (défaut), "02" ou "00" (le test d'installation).
dossier	Dossier où les exercices ont été installés (le dest de installer_exercices()). Par défaut "exercices-typst".
force	Logique. Réinitialiser sans confirmation interactive ? Par défaut FALSE (en mode non-interactif, force = TRUE est requis).

Value

Invisiblement, le chemin du dossier réinitialisé, ou NULL si l'opération a été annulée.

Examples

```
reinitialiser_exercice("01")
```

`valider_brand` *Valider un fichier* `_brand.yml`

Description

Vérifie qu'un `_brand.yml` est cohérent : schéma valide (via le paquet `brand.yml`), références croisées (couleurs, polices, logo) et existence des fichiers de polices `source: file`.

Usage

```
valider_brand(chemin = "_brand.yml")
```

Arguments

`chemin`	Chemin du fichier `_brand.yml`. Par défaut `"_brand.yml"`.

Value

Invisiblement, TRUE si tout est cohérent, FALSE sinon.

Examples

```
valider_brand()
```

`verifier_installation` *Vérifier que votre environnement est prêt pour le tutoriel*

Description

Contrôle, dans l'ordre, votre version de R, la présence et la version de Quarto, les paquets R prérequis, puis (par défaut) effectue un **rendu de test** d'un mini-document Typst pour valider la chaîne complète R -> Quarto -> Typst -> `gt` -> `ggplot2`. Affiche un bilan et la prochaine étape conseillée.

Usage

```
verifier_installation(tester_rendu = TRUE)
```

Arguments

`tester_rendu`	Logique. Effectuer le rendu de test Typst ? Activé par défaut. Mettez `FALSE` pour un contrôle rapide sans rendu.

Details

Le rendu de test utilise un fichier `format: typst` volontairement minimal (sans `_brand.yml` ni polices) : il valide la chaîne de compilation, mais pas la chaîne des polices de marque (le vrai point d'attention de l'exercice 2). Il fonctionne hors-ligne.

Value

Invisiblement, `TRUE` si tout est prêt, `FALSE` sinon.

Examples

```
verifier_installation()
verifier_installation(tester_rendu = FALSE)
```

Index