

# Initiation à Quarto

Marie Vaugoyeau

3 October 2023

## Table of contents

<b>C'est quoi Quarto ?</b>	<b>2</b>
<b>Création d'un projet Quarto</b>	<b>2</b>
<b>Lecture d'un fichier Quarto</b>	<b>3</b>
En-tête YAML (facultative) . . . . .	3
Syntaxe du texte . . . . .	3
Portion de code . . . . .	4
<b>Modification d'un fichier Quarto</b>	<b>4</b>
Modification de l'en-tête . . . . .	4
Utilisation des blocs d'interpellation ( <i>callout</i> ) . . . . .	5
Ajout d'un chunk de code . . . . .	7
<b>Réalisation d'un support de présentation</b>	<b>7</b>
Création de nouvelles diapos . . . . .	8
Incrementation des puces . . . . .	9
Faire plusieurs colonnes . . . . .	10
Ajout du scrolling pour les présentations <code>reactjs</code> . . . . .	11
<b>Conclusion</b>	<b>11</b>
<b>En savoir un peu plus sur moi</b>	<b>11</b>

*Ce support, produit pour le live du [3 octobre 2023 sur Twitch](#), est mis à disposition selon les termes de la [Licence Creative Commons Attribution 4.0 International](#).*

## C'est quoi Quarto ?



Après une dizaine d'année de développement sur **RMarkdown**, **Posit** (anciennement RStudio) a [annoncé en 2022](#) la naissance de **Quarto**.

Donc **Quarto** est l'héritier de **RMarkdown** qui combine le texte sous format **Markdown** et les portions de code appelé **chunk**.

Le code peut-être en **R** comme en **Python**.

Comme **RMarkdown**, sa grande force est de rendre **reproductible**, **répétable** et **réutilisable** les lignes de codes.

### Innovation

Il n'y aura pas ou peu d'innovations sur **RMarkdown** maintenant que **Quarto** existe.

## Création d'un projet Quarto

La grosse nouveauté entre **RMarkdown** et **Quarto** est que **Quarto** est une infrastructure plus large qui remplace le classique projet RStudio.

Pour créer un projet **Quarto**, c'est très simple, il suffit de cliquer sur l'icône bleue projet ou **File > New Project....**

Ensuite, il faut cliquer sur **Quarto project**. S'il n'apparaît pas ce que RStudio n'est pas à jour. Il est nécessaire de commencer par mettre à jour R et RStudio.

Le projet créé contient trois fichiers :

- \_ Le fichier `_quarto.yml` qui contient les paramètres du dossier
- \_ Le fichier `nom_du_projet.qmd` qui est un document **Quarto** généré automatiquement
- \_ Le fichier `nom_du_projet.Rproj`, le fichier du projet

Le document **Quarto** généré automatiquement est très simple. Il est là pour comprendre le fonctionnement de base. Il est très utile pour apprendre à manipuler **Quarto**.

## Lecture d'un fichier Quarto

Les fichiers Quarto ont une composition classique : une en-tête **YAML** facultative avec un enchaînement de **texte** et de **blocs de codes**.

Deux **vues** sont disponibles :

\_\_ **Visual** : Qui ressemble à ce qui sera obtenu avec des clics boutons proches d'un logiciel de traitement texte tels que **Word**, **Mot**, **Writer**...

\_\_ **Source** : Qui contient le script en **Markdown**

### En-tête YAML (facultative)

Elle permet de paramétrer les sorties du document mais elle n'est pas obligatoire

#### ! Indentation

Si l'indentation (les espaces avant le texte) n'a que peu d'importance dans le code **R**, elle est très importante en **YAML** et doit-être regardé avec attention !

Le document généré automatiquement ne possède qu'un titre dans l'en-tête.

### Syntaxe du texte

Le texte en Quarto, utilise la [syntaxe Markdown](#) :

\_\_ Les titres sont caractérisés par des **#** en fonction du niveau : **#Titre de niveau 1**, **##Titre de niveau 2**...

\_\_ La mise en forme du texte se fait avec les étoiles ou les impostrophes (apostrophe à l'envers) :

- Une étoile **\*** avant et après la partie à valoriser permet de mettre en *italique* (codé **\*italique\***)
- Deux **\*\*** avant et après mettent en **gras** (codé **\*\*gras\*\***)
- Trois **\*\*\*** avant et après mettent en ***gras et italique*** (codé **\*\*\*gras et italique\*\*\***)
- Un **double espace** permet de passer à la ligne. Il faut donc toujours mettre deux espaces à la fin de chaque ligne. Sans ces espaces, toutes les lignes sont collées les unes à la suite des autres

- Pour éditer un `format code` (sans qu'il se lance) pour présenter les packages, les fonctions ou les objets utilisés (comme fait dans ce document), il faut encadrer d'impostrophes

Des images peuvent aussi être intégrées grâce à `![*légende_de_l_image*](adresse_de_l_image.format_de_l_image)`

## Portion de code

Le code est enregistré dans des chunks délimités par trois impostrophes comme ci-dessous.

Les paramètres du chunk sont précédés de `#|`.

Cela permet de donner un nom au chunk avec `#| label: nom_du_chunk` ou de choisir les sorties `#| echo: true` : le code est affiché dans le format de sortie, `#| warning: false` : les messages de type `warning` ne sont pas affichés...

Tous les paramétrages peuvent se retrouver dans [le guide Quarto](#).

```
```${r}
#| label: nom_du_chunk
#| warning: false

1 + 1
```
```

[1] 2

Si un des impostrophes est supprimé par erreur, le code ne peut pas se lancer !

Il faut toujours vérifier que les chunks soient bien fermés (fond d'une couleur différente).

## Modification d'un fichier Quarto

### Modification de l'en-tête

Lors du Twitch, j'ai montré comment changer le titre et ajouter un.e auteur.trice dans l'en-tête.

```
---
title: "Initiation à *Quarto*"
author: "Marie Vaugoyeau"
---
```

Il est aussi possible de modifier le format de sortie pour générer un pdf avec un sommaire cliquable `toc: true` :

```

---
title: "Initiation à *Quarto*"
author: "Marie Vaugoyeau"
format:
  pdf:
    toc: true
---

```

Ou paramétrer différemment deux formats de sortie :

```

---
title: "Initiation à *Quarto*"
author: "Marie Vaugoyeau"
format:
  html:
    toc: true
    code-fold: true
    echo: false
    eval: false
  pdf:
    toc: true
---

```

#### Format de sortie

Pour connaître tous les formats de sorties générés par un fichier **Quarto**, rendez-vous sur [le guide de Quarto](#)

## Utilisation des blocs d'interpellation (*callout*)

J'ai aussi montrer comme ajouter des boîtes d'interpellations comme celles ci-dessous.

```

::: callout-tip
## Les statistiques

```

```

Ensemble de méthodes qui ont pour objet la collecte, le traitement et l'interprétation
:::

```

### Les statistiques

Ensemble de méthodes qui ont pour objet la **collecte**, le **traitement** et l'**interprétation** de l'ensemble des **données d'observation** relatives à une **population statistique** (groupe d'individus ou d'unités) ou concernant un phénomène quelconque.

```
::: callout-note
## Un bon code
```

```
Est un code qui produit les résultats attendus suffisamment vite et facile à maintenir
:::
```

### Un bon code

Est un code qui produit les **résultats attendus** suffisamment vite et **facile à maintenir**

```
::: callout-warning
## A retenir
```

```
Il faut réfléchir pour analyser les données de manière adaptée et interpréter correctement
:::
```

### A retenir

Il faut réfléchir pour analyser les données de **manière adaptée** et **interpréter correctement** les résultats

```
::: callout-important
## Erreurs à ne pas commettre
```

```
- r fort ne signifie pas une relation de cause à effet entre les deux variable\
- r faible ne signifie pas pas de relation entre les deux variables. Il peut y avoir un
:::
```

### Erreurs à ne pas commettre

- r fort ne signifie pas une **relation de cause à effet** entre les deux variable
- r faible ne signifie pas **pas de relation** entre les deux variables. Il peut y avoir

une relation non linéaire


```
::: {.callout-caution collapse="true"}  
## Mieux comprendre l'utilisation du développement
```

Ce bloc est un exemple qui ne se développe qu'en cliquant sur la flèche. Il faut utiliser ``c`  
:::

#### Mieux comprendre l'utilisation du développement

Ce bloc est un exemple qui ne se développe qu'en cliquant sur la flèche. Il faut utiliser `collapse="true"` comme cela `{.callout-tip/note/warning/important/caution collapse="true"}`.

## Ajout d'un chunk de code

Très simplement grâce au raccourci clavier `Ctrl + Alt + I` ou le clic bouton carré vert 

Il est aussi possible d'ajouter un calcul simple comme :

La longueur moyenne de longueur de sepal du jeu de données iris : 6 cm

Codé par `La longueur moyenne de longueur de sepal du jeu de données iris :  $r$   
mean(iris$Sepal.Length) /> round() cm avec des impostrophes autour du texte en italique.`

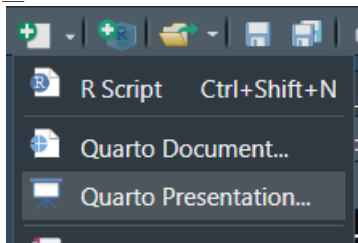
## Réalisation d'un support de présentation

Pour moi, `Quarto` est meilleur que `RMarkdown` pour générer des supports de présentations et comme vu précédemment, le même fichier peut-être généré en `html`, `revealjs`, `pdf`...

#### Pour faire un support de présentation

Le mieux est généré un nouveau fichier dédié via :

Un clic-bouton sur le rectangle blanc avec un plus sur fond vert



Dans le menu File > New File > Quarto Presentation...

Tous les choix faits dans la boîte de dialogue peuvent-être modifiés après

## Création de nouvelles diapos

De base, une nouvelle diapo est créé par titre de deuxième niveau **##**. Les titres de premier niveau donne des titres de section : **#**

```
---  
title: "Création d'un support"  
author: "Marie Vaugoyeau"  
format: revealjs  
---
```

**## Première diapo**

- Création d'une liste
- Contenant plusieurs objets

**## Deuxième diapo**

Ne contenant que du texte

```
::: {.callout-tip collapse="true"}  
## Et un **bloc d'interpellation**
```

Il en existe 5 sortes :

- ``note`` en bleu avec un ``i`` dans un rond par défaut
- ``warning`` en orange avec un ``!`` dans un triangle par défaut
- ``important`` en rouge avec un ``!`` dans un rond par défaut
- ``tip`` en vert avec une ampoule par défaut
- ``caution`` en jaune avec un cône de chantier par défaut



:::

Pour faire des diapos sans titre, il faut les séparer par ---

---

```
title: "Création d'un support"
author: "Marie Vaugoyeau"
format: revealjs
```

---

---

- Création d'une liste
- Contenant plusieurs objets

---

Ne contenant que du texte

```
::: {.callout-tip collapse="true"}
## Et un **bloc d'interpellation**
```

Il en existe 5 sortes :

- ``note`` en bleu avec un ``i`` dans un rond par défaut
- ``warning`` en orange avec un ``!`` dans un triangle par défaut
- ``important`` en rouge avec un ``!`` dans un rond par défaut
- ``tip`` en vert avec une ampoule par défaut
- ``caution`` en jaune avec un cône de chantier par défaut

:::

## Incrementation des puces

Les puces peuvent apparaître pas à pas grâce à l'option `incremental` soit sur un bloc :

```
::: {.incremental}
```

- Création d'une liste
- Contenant plusieurs objets

```
:::
```

```
::: {.nonincremental}
```

- Création d'une liste
- Contenant plusieurs objets

```
:::
```

Soit sur tous le document grâce à l'option dans l'en-tête (d'où l'intérêt du `nonincremental` d'au-dessus).

```
format:
```

```
  revealjs:
```

```
    incremental: true
```

## Faire plusieurs colonnes

La création de colonnes est possible diapos, par diapos grâce à l'option `{.columns}`

```
::: {.columns}
```

```
::: {.column width="40%"}
```

```

```

```
:::
```

```
::: {.column width="60%"}
```

Bonjour,

Je suis **Marie VAUGOYEAU**, accompagnatrice indépendante à l'analyse de données

Ce que je préfère faire **c'est** aider les chercheu.se.r.s dans la valorisation de leurs données

En plus de mes missions, j'aime partager mes connaissances en statistiques et R, en partie sur

J'ai aussi écrit [un livre](<https://www.editions-eni.fr/livre/langage-r-et-statistiques-initi>).

```
:::
```

```
::::
```

## Ajout du scrolling pour les présentations reactjs

Lorsqu'il y a beaucoup d'information sur une même diapo, le texte ne peut pas être affiché en entier. En effet, **la taille du texte ne varie pas en fonction de la place sur la page**.

Plutôt que de réduire la taille de la police, il est possible d'ajouter du **scrolling** soit pour une diapo sur la ligne du titre `## Titre de la diapo {.scrollable}` soit dans l'en-tête pour toutes les diapos

```
format:
  revealjs:
    incremental: true
    scrollable: true
```

## Conclusion

Impossible de parler de toutes les fonctionnalités possible, j'espère que cet article permettra à ceratin.e.s de se lancer dans l'utilisation de Quarto ! Pour en savoir plus, rendez-vous sur [la page dédiée](#) !

## En savoir un peu plus sur moi

Bonjour,

Je suis Marie Vaugoyeau et je suis disponible pour des **missions en freelance d'accompagnement à la formation** à R et à l'analyse de données et/ou en **programmation** (reprise de scripts, bonnes pratiques de codage, développement de package).

Ayant un **bagage recherche en écologie**, j'ai accompagné plusieurs chercheuses en biologie dans leurs analyses de données mais je suis ouverte à d'autres domaines.

Vous pouvez retrouver mes offres [ici](#).

**En plus de mes missions de consulting je diffuse mes savoirs en R et analyse de données sur plusieurs plateformes :**

- J'ai écrit [un livre aux éditions ENI](#)
- Tous les mois je fais [un live sur Twitch](#) pour parler d'un package de R, d'une analyse
- Je rédige une **newsletter** de manière irrégulière pour parler de mes **inspirations** et transmettre **des trucs et astuces sur R**. Pour s'y inscrire, [c'est par là](#). J'ai aussi [un blog](#) sur lequel vous pourrez retrouver une version de cet article.

Pour en savoir encore un peu plus sur moi, il y a [LinkedIn](#) et pour retrouver [tous ces liens et plus encore, c'est ici](#)

N'hésitez pas à me contacter sur [marie.vaugoyeau@gmail.com](mailto:marie.vaugoyeau@gmail.com) !

Bonne journée

Marie

