

# TYPESCRIPT

## ASTUCES AVEC L'OPÉRATEUR typeof

# OPÉRATEUR `typeof` EN JAVASCRIPT

- Usage : réflexion au runtime
- Limité → types primitifs, `function`, `object`

# OPÉRATEUR `typeof` EN TYPESCRIPT

- S'appuie sur les capacités d'inférence de types du compilateur TypeScript
- Permet de définir des types au compile time

→ *Cas d'utilisation ?*

# API REST + SWAGGER

👉 <https://petstore.swagger.io/#/pet/addPet>

Donne exemples de valeurs JSON en entrée/sortie

→ *Comment utiliser ses samples pour en déduire des types utilisables en TypeScript ?*

## CONVERTISSEUR JSON → TS

 <http://www.json2ts.com/>

Inconvénients :

- Nécessite une action manuelle intermédiaire
- Découpage fin pas forcément utile

## JSON → JS + `typeof` → TS



👉 <http://www.typescriptlang.org/play/>

- Définir une constante `petSample` avec la valeur
  - Elle porte en elle le type - *cf. IntelliSense*
- Définir un type associé :
  - `type PetDto = typeof petSample;`

## AVANTAGES

- Marche aussi pour les types imbriqués :
  - `type TagSample = typeof petSample.tags[0];`
- Réutilisable ailleurs :
  - `httpClient.get<PetDto>(...)`

## LIMITES (1)


- Nom du type pas utilisé par IntelliSense
- Contournement : `interface` vide, juste étendant le type
- Inconvénients :
  - Etape en plus
  -  Interface n'est pas "close" en TypeScript
  -  Interface pour le sous-type



## LIMITES (2)

- Option `strictNullChecks` activée => champ non nullable, non optionnel par défaut
- Contournements :
  - Pour rendre tous les champs optionnels : type `Partial<T>`
  - Pour rendre un champ optionnel ou nullable :
    - 💡 Wrapper le champ dans une fonction `nullable`
    - ⚠ Risque de sauter à la prochaine version de l'API

## CONSEIL

 *Dès personnalisation plus fine,  
écrire directement les type alias ou les  
interfaces sans passer par des  
samples.*

## LIMITES GLOBALES DE CES TYPES

- Purement TypeScript / compile-time.
  - 0 impact sur le JavaScript compilé.
- Pour vérifier les types au runtime
  - <https://github.com/gcanti/io-ts>
    - Artillerie lourde <-> API untrusty

## POUR ALLER PLUS LOIN

- Interface vs Type alias :  
[https://medium.com/@martin\\_hotell/2a8f1777af4c](https://medium.com/@martin_hotell/2a8f1777af4c)
- Typescript Type Inference Guide :  
<http://ducin.it/typescript-type-inference-guide>
- TypeScript—Make types “real”, the type guards :  
<https://medium.com/@wittydeveloper/814364e8dbe3>