

String / int / bool en Dart

Table des matières

I. Type string	3
II. Exercice : Quiz	6
III. Type int	7
IV. Exercice : Quiz	8
V. Type bool	9
VI. Exercice : Quiz	11
VII. Essentiel	12
VIII. Auto-évaluation	12
A. Exercice	12
B. Test	12
Solutions des exercices	13

I. Type string

Contexte

Un langage de programmation possède en général plusieurs types d'objets qui permettent de manipuler des types de données différents. Ceux-ci sont indispensables en Dart pour définir ce que contient une variable ou encore ce que retourne une fonction.

Parmi les nombreux types d'objets qui existent en Dart, trois sont largement utilisés : **String**, **int** et **bool**. **String** correspond aux chaînes de caractères, **int** correspond aux nombres entiers et **bool** aux valeurs booléennes.

Dans la plupart des autres langages de haut niveau, on retrouve également ces types de données. Il est donc indispensable de savoir les manipuler, de connaître leurs propriétés et d'utiliser les méthodes adaptées.

À travers ce cours, nous allons définir ces types, voir comment les déclarer et leur assigner une valeur, analyser leurs propriétés et méthodes, et enfin mettre tout cela en application.

Définition

Le type **String** correspond à une chaîne de caractères (suite ordonnée de caractères). Il est utilisé principalement pour représenter un texte.

Pour rentrer dans les détails, un objet de type **String** contient une séquence d'unités de code UTF-16, qui est un codage de caractères largement utilisé.

Fondamental Déclaration et assignation

Voici comment déclarer une variable de type **String** :

```
// La valeur "Hello" est assignée à la variable a.
var a = "Hello";
// La variable a est donc égale à "Hello" après l'opération d'assignation.
// Il est également possible d'assigner une valeur avec des guillemets
simples :
var b = 'Hello';
// Si l'on affiche la valeur de a ou b dans la console, on obtient : Hello.
// Il est possible d'intégrer des guillemets simples ou doubles dans une
chaîne de caractères, en mélangeant les deux :
var c = "`Hello`";
// Si l'on affiche la valeur de c dans la console, on obtient : 'Hello'.
var d = `"Hello"`;
// Si l'on affiche la valeur de d dans la console, on obtient : "Hello".
```

Fondamental Propriétés et méthodes utiles

Les propriétés et méthodes ci-dessous seront mises en application dans la sous-partie suivante.

Voici les propriétés à connaître concernant le type **String** :

- Pour mettre la valeur d'une expression à l'intérieur d'un objet String (par exemple, la valeur d'une variable), on utilise `${expression}` (on appelle cela l'interpolation).
- L'opérateur « + » est utilisé pour concaténer (additionner) des chaînes de caractères.
- Pour accéder à un caractère précis d'un objet String, on utilise un index (exemple : `string[0]`).
- L'attribut `length` (`string.length`) permet d'obtenir la longueur de la chaîne de caractères.

Il existe également plus d'une trentaine de méthodes attachées au type **String**. Voici les plus utilisées d'entre elles :

- **IndexOf(pattern)** : retourne la position de la première correspondance avec la chaîne de caractères *pattern* (permet en fait de tester si l'objet String contient une sous-chaîne de caractères).
- **ReplaceAll(pattern, replace)** : remplace toutes les occurrences de *pattern* au sein de l'objet String par la sous-chaîne de caractères *replace*.
- **Split(pattern)** : divise la chaîne de caractères selon le séparateur *pattern* et renvoie un tableau de String avec les sous-chaînes séparées.
- **Substring(start, end)** : coupe la chaîne de caractères et ne conserve que ce qui est contenu entre l'index *start* et l'index *end*.
- **Trim()** : retourne la chaîne de caractères en enlevant les espaces blancs au début et à la fin de celle-ci.

Exemple Mise en pratique

Exemples pour les différentes propriétés d'un objet **String** :

```
// Les valeurs "Hello" et "World" sont assignées à deux variables :
var a = "Hello";
var b = 'World';

// Notre objectif est de créer un objet string : Hello World.
// Voici la première méthode en utilisant l'interpolation :
var c = "${a} ${b}";

// Voici la deuxième méthode en utilisant la concaténation :
var d = a + " " + b;

// Les variables c et d contiennent toutes deux la valeur "Hello World".
// Nous voulons maintenant accéder au troisième caractère de la
variable c.
var e = c[2];

// Et non c[3], car les index commencent toujours par 0 et non 1.
// La variable c contient alors la valeur 1.
// Enfin, nous souhaitons connaître la longueur de la variable d.
var f = d.length;

// La variable f contient la valeur 11.
```

Exemple pour la méthode *indexOf* d'un objet **String** :

```
// Nous assignons une valeur à la variable a.
var a = "Bonjour, comment allez-vous ?";
// Nous utilisons la méthode indexOf pour définir si la variable a contient
le mot "comment".
var b = a.indexOf("comment");
// b contient alors la valeur 9, car le mot commence à la 9e position.
var c = a.indexOf("Hello");
// c contient alors la valeur -1, ce qui signifie que la valeur recherchée
n'est pas présente dans la variable a.
```

Exemple pour la méthode *replaceAll* d'un objet **String** :

```
// Nous assignons une valeur à la variable a.
var a = "Bonjour, comment allez-vous ?";
// Nous utilisons la méthode replaceAll pour remplacer certaines sous-
chaînes.
var b = a.replaceAll("ou", "on");
// b contient alors la valeur "Bonjonr, comment allez-vons ?"
```

Exemple pour la méthode *split* d'un objet **String** :

```
// Nous assignons une valeur à la variable a.
var a = "Bonjour, comment allez-vous ?";
// Nous utilisons la méthode split pour diviser la chaîne de caractères
suivant le séparateur "ou".
var b = a.split("ou");
// b contient alors le tableau de string suivant :
// [ "Bonj",
// "r, comment allez-v",
// "s ?"
// ]
```

Exemple pour la méthode *substring* d'un objet **String** :

```
// Nous assignons une valeur à la variable a.
var a = "Bonjour, comment allez-vous ?";
// Nous utilisons la méthode subString pour extraire la partie comprise
entre l'index 9 (inclus) et l'index 27 (exclu).
var b = a.substring(9, 27);
// b contient alors la valeur "comment allez-vous".
```

Exemple pour la méthode *trim* d'un objet **String** :

```
// Nous assignons une valeur à la variable a.
var a = " Bonjour, comment allez-vous ? ";
// Nous utilisons la méthode trim pour enlever tous les espaces blancs.
var b = a.trim();
// b contient alors la valeur "Bonjour, comment allez-vous ?".
```

Exercice : Quiz

[solution n°1 p.15]

Question 1

Quel type de données contient un objet String ?

- ☐ Un caractère ASCII
- ☐ Une chaîne de caractères

Question 2

Quelle réponse ci-dessous est correcte pour déclarer une variable String ?

- ☐ var a = 'Hello'
- ☐ var a = Hello
- ☐ var a = "Hello"

Question 3

Il existe une propriété permettant de récupérer la longueur d'un objet String.

- ☐ Vrai
- ☐ Faux

Question 4

Quelle méthode utilise-t-on pour tester si un objet string contient une sous-chaîne de caractères ?

- ☐ Split
- ☐ IndexOf
- ☐ ReplaceAll

Question 5

La méthode *trim* permet d'enlever les espaces blancs au milieu d'un objet String.

- ☐ Vrai
- ☐ Faux

III. Type int

Définition

Le type **int** permet de représenter un nombre entier. Les entiers sont des nombres sans partie décimale. Ce nombre doit être compris entre -2^{63} et $2^{63} - 1$ (ce qui est extrêmement large) car il est contenu sur 64 bits.

Fondamental Déclaration et assignation

Voici comment déclarer une variable de type **int** :

```
// La valeur 3 est assignée à la variable a.  
var a = 3;  
  
// La variable a est donc égale à 3 après l'opération d'assignation.  
  
// Si la valeur assignée est une constante qui ne sera jamais modifiée,  
// on peut utiliser le mot-clé const.  
const b = 3;  
  
// Il est également possible de déclarer un nombre hexadécimal avec le  
// préfixe 0x.  
var c = 0xDEAF09F1;
```

Fondamental Propriétés et méthodes utiles

Les propriétés et méthodes ci-dessous seront mises en application dans la sous-partie suivante.

Voici les propriétés à connaître concernant le type **int** :

- Pour savoir si le nombre est pair ou impair, on peut utiliser les propriétés *isEven* (pair) ou *isOdd* (impair) qui retournent *true* ou *false*.
- Pour connaître le signe de l'entier (+ ou -), on peut utiliser la propriété *sign*, qui retourne -1 pour une valeur négative, +1 pour une valeur positive et 0 pour une valeur égale à 0.

Voici les méthodes à connaître concernant le type **int** :

- **Abs()** : retourne la valeur absolue du nombre entier,
- **Truncate()** : retourne la partie entière d'un nombre,
- **Round()** : retourne l'entier le plus proche d'un nombre,
- **ToString()** : transforme un nombre en une chaîne de caractères.

Exemple Mise en pratique

Exemples pour les différentes propriétés d'un objet **int** :

```
// La valeur -2 est assignée à la variable a.
var a = -2;

// On cherche à savoir si le nombre est pair avec la méthode isEven.
var b = a.isEven;

// La variable b est égale à true, car -2 est bien un nombre pair.

// On cherche à savoir si le nombre est impair avec la méthode isOdd.
var c = a.isOdd;

// La variable c est égale à false, car -2 n'est pas un nombre impair.

// On cherche à connaître le signe de l'entier compris dans la variable a.
var d = a.sign;

// La variable d est égale à -1, car -2 est une valeur négative.
```

Exemples pour les différentes méthodes d'un objet **int** :

```
// La valeur -2 est assignée à la variable a.
var a = -2;

// On cherche à obtenir la valeur absolue de a.
var b = a.abs();

// La variable b est égale à 2, car il s'agit de la valeur absolue de -2.

// On déclare un nombre qui n'est pas un nombre entier.
var c = 1.75;

// On cherche à obtenir la partie entière de la variable c.
var d = c.truncate();

// La variable d est égale à 1, car il s'agit de la partie entière de 1.75.

// On cherche à obtenir l'entier le plus proche de la variable c.
var e = c.round();

// La variable e est égale à 2, car il s'agit de l'entier le plus proche de 1.75.

// On cherche à transformer la variable e en chaîne de caractères (String).
var f = e.toString();

// La variable f contient la chaîne de caractères "2".
```

Exercice : Quiz

[solution n°2 p.16]

Question 1

Quel type de données contient un objet int ?

- ☐ Un nombre réel
- ☐ Un nombre entier
- ☐ Un chiffre compris entre 0 et 9

Question 2

Quelle réponse ci-dessous est correcte pour déclarer une variable int ?

- ☐ var a = 2
- ☐ var a = "2"
- ☐ var a = deux

Question 3

Il existe une propriété permettant de savoir si l'entier est pair ou impair.

- ☐ Vrai
- ☐ Faux

Question 4

Quelle méthode utilise-t-on pour déterminer la valeur absolue d'un nombre entier ?

- ☐ Round
- ☐ ToString
- ☐ Abs

Question 5

La méthode *truncate* permet de ne conserver que la partie décimale d'un nombre.

- ☐ Vrai
- ☐ Faux

V. Type bool

Définition

Le type **bool** permet de représenter une valeur booléenne.

Une valeur booléenne ne peut représenter que deux informations : VRAI ou FAUX. VRAI est représentée par le mot-clé *true* et FAUX est représentée par le mot-clé *false*.

Ce type de données est par ailleurs souvent utilisé avec les expressions booléennes lors des tests conditionnels (boucle *for*, *while*, condition *if*).

Fondamental Déclaration et assignation

Voici comment déclarer une variable de type **bool** :

```
// La valeur false est assignée à la variable a.
var a = false;
// La variable a est donc égale à false après l'opération d'assignation.
```

Fondamental Propriétés et méthodes utiles

Les propriétés et méthodes ci-dessous seront mises en application dans la sous-partie suivante.

Voici les propriétés à connaître concernant le type **bool** :

- Lorsque l'on utilise l'opérateur « == » (opérateur pour tester une égalité), une valeur de type **bool** est renvoyée.
- Pour inverser une expression booléenne (passer de VRAI à FAUX, ou de FAUX à VRAI), on utilise l'opérateur « ! ».
- Pour cumuler des expressions booléennes, on utilise les opérateurs « && » (opération logique ET) et « || » (opération logique OU).

Il n'y a qu'une méthode attachée au type **bool** :

- **ToString()** : permet de convertir la valeur en une chaîne de caractères (*true* devient "true" et *false* devient "false").

Exemple Mise en pratique

Exemples pour les différentes propriétés d'un objet **bool** :

```
// Des nombres entiers sont assignés aux variables a et b.
var a = 2;
var b = 3;

// Une opération d'égalité est effectuée et retourne donc une valeur booléenne.
var d = a == b;

// La variable d est égale à false (FAUX), car les valeurs de a et b ne sont pas égales.

// On inverse la variable d avec l'opérateur « ! ».
var e = !d;

// La variable e est donc égale à true (VRAI).

// On cumule les variables booléennes avec l'opérateur « && ».
var f = d && e;

// On cumule les variables booléennes avec l'opérateur « || ».
var g = d || e;

// La variable f est égale à false, car les variables d ET e ne sont pas toutes les deux égales à true.

// La variable g est égale à true, car la variable d OU la variable e est au moins égale à true (il s'agit de la variable e dans ce cas).
```

Exemple de la méthode *toString* d'un objet **bool** :

```
// La valeur true est assignée à la variable a.  
var a = true;  
  
// On utilise la méthode toString pour transformer la valeur booléenne  
en String.  
var b = a.toString();  
  
// La variable b contient la valeur "true" (chaîne de caractères).
```

Exercice : Quiz

[solution n°3 p.17]

Question 1

Quelles valeurs peut prendre un objet bool ?

- ☐ 0 et 1
- ☐ True ou false
- ☐ Vrai ou faux

Question 2

Quelle réponse ci-dessous est correcte pour déclarer une variable booléenne ?

- ☐ var a = vrai
- ☐ var a = "true"
- ☐ var a = false

Question 3

Pour inverser une expression booléenne, on utilise l'opérateur « ! ».

- ☐ Vrai
- ☐ Faux

Question 4

Quelle est la seule méthode disponible pour un objet bool ?

- ☐ Round
- ☐ Split
- ☐ ToString

Question 5

On peut cumuler des expressions booléennes avec l'opérateur « && ».

- ☐ Vrai
- ☐ Faux

VII. Essentiel

Les types **String**, **int** et **bool** sont les types d'objets les plus couramment utilisés dans le langage Dart. Que ce soit pour contenir un texte, un nombre ou une valeur booléenne, on comprend vite qu'il est indispensable de les connaître sur le bout des doigts.

Ils sont accompagnés de diverses propriétés et méthodes qui facilitent leur manipulation. Que ce soit pour récupérer la longueur d'une chaîne de caractères, tronquer un nombre entier ou encore cumuler des expressions booléennes, ce sont là des opérations que l'on utilise quotidiennement.

Dans ce cours, nous avons analysé et mis en application les principales connaissances à avoir concernant ces types de données. Lorsqu'un doute subsiste ou que vous souhaitez avoir davantage de détails concernant une propriété ou une méthode, vous pouvez consulter la documentation fournie sur le site Dart SDK¹.

Avec la pratique, la manipulation des objets de type **String**, **int** et **bool** deviendra familière, tant ils sont présents dans le quotidien du développeur.

VIII. Auto-évaluation

A. Exercice

Dans cet exercice, nous allons utiliser et mettre en application les différents types d'objets vus dans ce cours. Les réponses doivent être ajoutées successivement à un programme en Dart que vous devez exécuter (pour simplifier l'environnement, le DartPad peut être utilisé : DartPad²).

Question 1

[solution n°4 p.18]

Déclarez une chaîne de caractères comprenant "Hello world".

Question 2

[solution n°5 p.18]

Affichez dans la console le caractère présent à l'index 7 de cette chaîne.

Question 3

[solution n°6 p.18]

Utilisez la méthode de votre choix pour remplacer le mot "Hello" de cette chaîne de caractères par le mot "Hi". Affichez le résultat dans la console.

Question 4

[solution n°7 p.18]

Utilisez la méthode de votre choix pour ne garder que la partie de la chaîne comprenant le mot "Hi". Affichez le résultat dans la console.

Question 5

[solution n°8 p.18]

Déclarez un entier égal à -5 et utilisez la méthode correspondante pour savoir si le nombre est pair. Affichez le résultat dans la console.

Question 6

[solution n°9 p.19]

Affichez la valeur absolue de ce nombre dans la console, en utilisant la méthode correspondante.

Question 7

[solution n°10 p.19]

Déclarez une variable booléenne égale à *false*, convertissez-la en *String* puis affichez le résultat dans la console.

B. Test

Exercice 1 : Quiz

[solution n°11 p.21]

Question 1

¹ <https://api.dart.dev/stable/2.9.0/index.html>

² https://dartpad.dev/?null_safety=true

Les types String, int et bool sont utilisés par le développeur :

- ☐ Quotidiennement
- ☐ De temps à autre, quand le code l'impose
- ☐ Rarement

Question 2

Le type String met à disposition un ensemble de méthodes facilitant la manipulation des chaînes de caractères.

- ☐ Vrai
- ☐ Faux

Question 3

Le type int est le seul type de données disponibles pour représenter un nombre en Dart.

- ☐ Vrai
- ☐ Faux

Question 4

Les expressions booléennes sont principalement utilisées :

- ☐ Lors des tests de conditions
- ☐ Pour effectuer des opérations mathématiques
- ☐ Pour représenter des textes

Question 5


Que faut-il pour se familiariser avec ces types de données et les maîtriser ?

- ☐ Les apprendre par cœur
- ☐ Pratiquer concrètement avec des exercices de code
- ☐ Relire plusieurs fois la documentation Dart au complet

Solutions des exercices


Exercice p. 6 Solution n°1**Question 1**

Quel type de données contient un objet String ?

- ☐ Un caractère ASCII
- ☒ Une chaîne de caractères
-  Le type String correspond à une chaîne de caractères, et il est principalement utilisé pour représenter un texte.


Question 2

Quelle réponse ci-dessous est correcte pour déclarer une variable String ?

- ☐ var a = 'Hello'
- ☐ var a = Hello
- ☒ var a = "Hello"
-  La chaîne de caractères doit être écrite entre guillemets (simples ou doubles).


Question 3

Il existe une propriété permettant de récupérer la longueur d'un objet String.

- ☒ Vrai
- ☐ Faux
-  Il s'agit de la propriété *length*, très souvent utilisée.


Question 4

Quelle méthode utilise-t-on pour tester si un objet string contient une sous-chaîne de caractères ?

- ☐ Split
- ☒ IndexOf
- ☐ ReplaceAll
-  *IndexOf* retourne la position de la première correspondance avec la sous-chaîne de caractères, et retourne - 1 si elle n'est pas présente.

Question 5


La méthode *trim* permet d'enlever les espaces blancs au milieu d'un objet String.

- ☐ Vrai
- ☒ Faux
-  La méthode *trim* enlève les espaces blancs au début et à la fin d'un objet String.

Exercice p. 8 Solution n°2


Question 1

Quel type de données contient un objet int ?

- ☐ Un nombre réel
- ☒ Un nombre entier
- ☐ Un chiffre compris entre 0 et 9
-  Le type int permet de représenter un nombre entier.


Question 2

Quelle réponse ci-dessous est correcte pour déclarer une variable int ?

- ☒ var a = 2
- ☐ var a = "2"
- ☐ var a = deux
-  Un entier est simplement défini en écrivant le nombre correspondant, sans guillemets.


Question 3

Il existe une propriété permettant de savoir si l'entier est pair ou impair.

- ☒ Vrai
- ☐ Faux
-  Il s'agit des propriétés *isOdd* (impair) et *isEven* (pair).


Question 4

Quelle méthode utilise-t-on pour déterminer la valeur absolue d'un nombre entier ?

- ☐ Round
- ☐ ToString
- ☒ Abs
-  La méthode *abs* retourne la valeur absolue d'un nombre entier.

Question 5


La méthode *truncate* permet de ne conserver que la partie décimale d'un nombre.

- ☐ Vrai
- ☒ Faux
-  La méthode *truncate* permet de ne conserver que la partie entière d'un nombre, et permet ainsi de récupérer un entier à partir d'un nombre décimal en sectionnant ce qui se situe après la virgule.

Exercice p. 11 Solution n°3


Question 1

Quelles valeurs peut prendre un objet bool ?

- ☐ 0 et 1
- ☒ True ou false
- ☐ Vrai ou faux
-  True et false sont les deux seules valeurs que peut prendre un objet bool.


Question 2

Quelle réponse ci-dessous est correcte pour déclarer une variable booléenne ?

- ☐ var a = vrai
- ☐ var a = "true"
- ☒ var a = false
-  Une valeur booléenne est définie en utilisant les mots-clés *true* ou *false* sans guillemets.


Question 3

Pour inverser une expression booléenne, on utilise l'opérateur « ! ».

- ☒ Vrai
- ☐ Faux
-  En plaçant un point d'exclamation devant une expression booléenne, cela a pour effet d'inverser sa valeur.


Question 4

Quelle est la seule méthode disponible pour un objet bool ?

- ☐ Round
- ☐ Split
- ☒ ToString
-  *ToString* permet de convertir la valeur booléenne en une chaîne de caractères.

Question 5

On peut cumuler des expressions booléennes avec l'opérateur « && ».

- ☒ Vrai
- ☐ Faux
-  L'opérateur « && » permet de combiner des expressions booléennes suivant l'opérateur logique « ET ».

p. 12 Solution n°4

```
// On assigne la valeur "Hello world" à la variable a.
var a = "Hello world";
```

p. 12 Solution n°5

```
// On affiche le caractère situé à l'index 7 de la chaîne de
caractères.
print(a[7]);
// Cela affiche le caractère o.
```

p. 12 Solution n°6

```
// On utilise la méthode replaceAll pour remplacer les mots
concernés.
var b = a.replaceAll("Hello", "Hi");
print(b);
// Cela affiche dans la console : Hi world.
```

p. 12 Solution n°7

```
// On utilise la méthode substring pour couper la string à l'index 2.
var c = b.substring(0, 2);
// Il est également possible d'utiliser la méthode split pour séparer
la String au niveau de l'espace blanc, puis d'en récupérer le
premier élément.
// var c = b.split(" ")[0];
print(c);
// Cela affiche dans la console : Hi.
```

p. 12 Solution n°8

```
// On assigne la valeur -5 à la variable d.
var d = -5;
// On utilise la méthode isEven pour savoir si le nombre est pair.
print(d.isEven);
// Cela affiche dans la console : false.
```

p. 12 Solution n°9

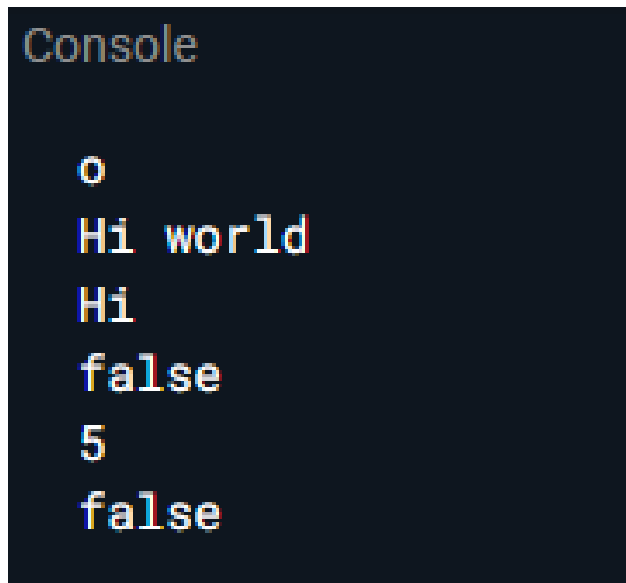
```
// On utilise la méthode abs pour retourner la valeur absolue du
nombre.
print(d.abs());
// Cela affiche dans la console : 5.
```

p. 12 Solution n°10

```
// On assigne la valeur false à la variable e.
var e = false;
// On utilise la méthode toString pour la convertir en string.
print(e.toString());
// Cela affiche dans la console : false.
```

Exemple de programme final :

```
void main() {  
    var a = "Hello world";  
  
    print(a[7]);  
  
    var b = a.replaceAll("Hello", "Hi");  
  
    print(b);  
  
    var c = b.substring(0, 2);  
    // var c = b.split(" ")[0];  
  
    print(c);  
  
    var d = -5;  
  
    print(d.isEven);  
  
    print(d.abs());  
  
    var e = false;  
  
    print(e.toString());  
}
```

Résultat affiché dans la console :

```
Console

o
Hi world
Hi
false
5
false
```

Exercice p. 12 Solution n°11**Question 1**

Les types String, int et bool sont utilisés par le développeur :

- ☒ Quotidiennement
- ☐ De temps à autre, quand le code l'impose
- ☐ Rarement
- ☐ Ces trois types de données sont constamment utilisés dans les programmes informatiques codés en Dart.

Question 2

Le type String met à disposition un ensemble de méthodes facilitant la manipulation des chaînes de caractères.

- ☒ Vrai
- ☐ Faux
- ☐ Un objet String dispose de nombreuses méthodes : indexOf, split, substring, replaceAll, etc.


Question 3

Le type int est le seul type de données disponibles pour représenter un nombre en Dart.

- ☐ Vrai
- ☒ Faux
- ☐ Le type int représente uniquement les nombres entiers, mais il existe d'autres types pour représenter les nombres décimaux par exemple.


Question 4

Les expressions booléennes sont principalement utilisées :

- ☒ Lors des tests de conditions
- ☐ Pour effectuer des opérations mathématiques
- ☐ Pour représenter des textes
-  Les opérateurs d'égalité (« == » ou « != ») ou encore de cumul d'expressions booléennes (« && » ou « || ») fonctionnent avec des expressions booléennes pour valider ou non des conditions données.

Question 5

Que faut-il pour se familiariser avec ces types de données et les maîtriser ?

- ☐ Les apprendre par cœur
- ☒ Pratiquer concrètement avec des exercices de code
- ☐ Relire plusieurs fois la documentation Dart au complet
-  La pratique est le meilleur entraînement pour progresser en Dart.