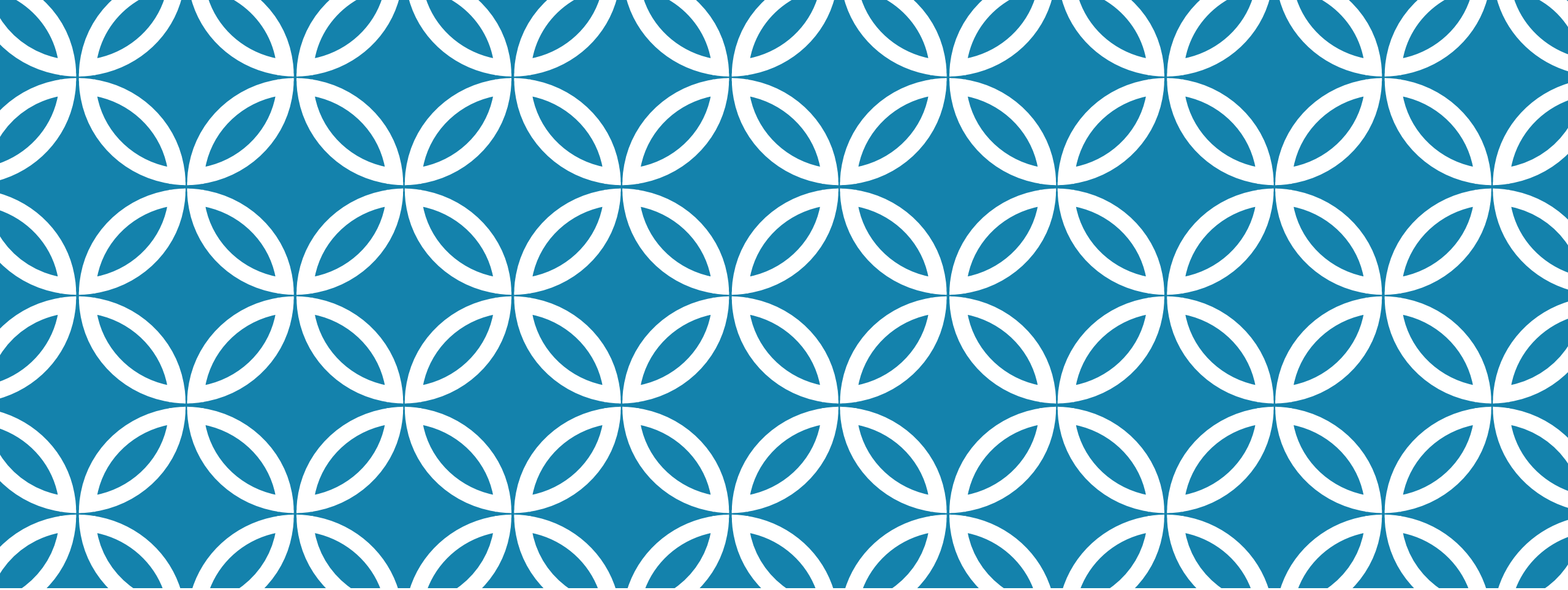
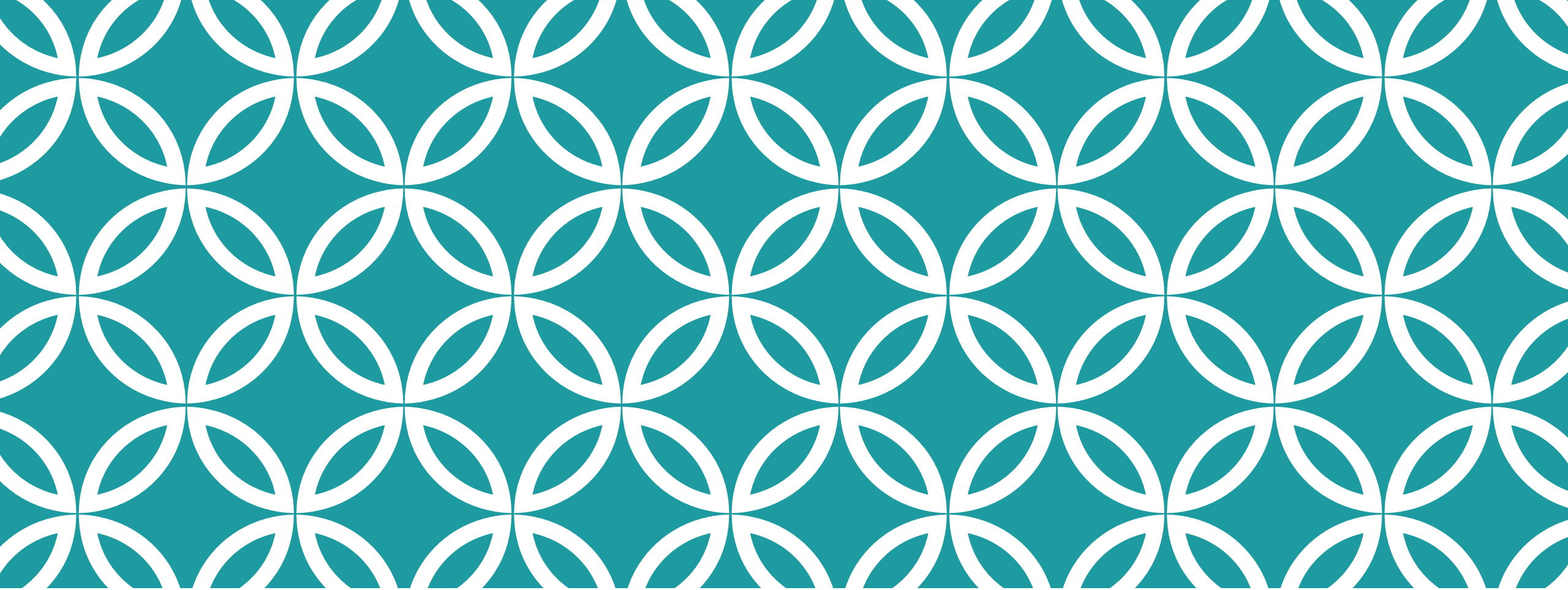


ALGORITHMIE

Jérémy PERROUAULT



INTRODUCTION OBJET

Introduction à l'objet

PROBLÉMATIQUE

Projection de la réalité difficile, on aimerait pouvoir regrouper des entiers, des chaines de caractères, des booléens, … Bref, regrouper des infos pour en constituer une structure plus ou moins complexe

- Et tant qu'à y être, les ordonner de façon logique

```
Entier[] lesAges;
Char[][] lesNoms;
Char[][] lesPrenoms;
```

Réutiliser des algorithmes écrits par d'autres (ou pas) de façon simplifiée

- Pourquoi pas masquer une complexité ?

IDÉE

Créer un type de variable (appelé « Structure ») qui s'utilisera comme un type classique

- On utilise le mot-clé « nouvelle » ou « nouveau » pour créer une nouvelle « Structure »
- Toutes les variables dans la structure sont accessibles publiquement
- L'accès se fait via le point (« . ») sur une variable de type « Structure »

```
Structure Personne {
  Caractere[] prenom;
  Caractere[] nom;
  Entier age = 0;
}
```

```
Personne maPersonne = nouvelle Personne();

maPersonne.age = 29;
```

EN PRATIQUE

```
Structure Personne {
  Caractere[] prenom;
  Caractere[] nom;
  Entier age = 0;
}
```

```
Personne jeremy = nouvelle Personne();

jeremy.age = 29;

Personne alissa = nouvelle Personne();
Personne noemie;
```

Adresse mémoire	Variable	Valeur
@1	jeremy.prenom	
@4	jeremy.nom	
@11	jeremy.age	29
@6	alissa.prenom	
@9	alissa.nom	
@10	alissa.age	0
	noemie	NULL

@1	@2	@3	@4	@5
@6	@7	@8	@9	@10
@11				

SPÉCIALISATION

```
Programme Booléen isPersonneChild(Personne p) {
  Si (p.age < 18) {
    Retour Vrai;
  }
  Retour Faux;
}
```

SPÉCIALISATION

Et si on allait plus loin ? Si on créait une structure qui a

- Ses propres données (comme c'est le cas actuellement)
- Ses propres fonctionnalités, son propre comportement ?

Et on appellerait ça une « classe »

- Et ça encapsulerait (masquerait) toute la complexité interne du fonctionnement de la classe
  - On gagne en intégrité et en sécurité

IDÉE

Créer un type de variable (appelé « Classe ») qui s'utilisera comme un type classique

- Et qui en plus d'avoir ses propres données, aura son comportement
- Utilisation du mot-clé « moi » dans une classe pour faire référence à ses propres données ou programmes

```
Classe Personne {
  Caractere[] prenom;
  Caractere[] nom;
  Entier age = 0;

  Programme Booléen isChild() {
    Si (moi.age < 18) {
      Retour Vrai;
    }
    Retour Faux;
  }
}
```

```
Personne maPersonne = nouvelle Personne();

maPersonne.age = 29;

Si (maPersonne.isChild()) {
  ecrire("C'est un enfant !!")
}
```

EN PRATIQUE

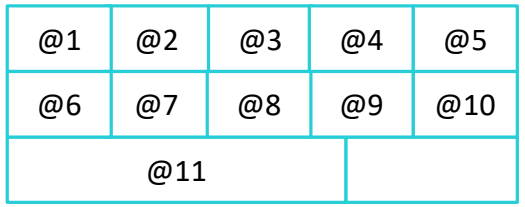
```
Classe Personne {
  Caractere[] prenom;
  Caractere[] nom;
  Entier age = 0;

  Programme Booléen isChild() {
    Si (moi.age < 18) {
      Retour Vrai;
    }
    Retour Faux;
  }
}
```

```
Personne maPersonne = nouvelle Personne();

maPersonne.age = 29;

Si (maPersonne.isChild()) {
  ecrire("C'est un enfant !!")
}
```



Adresse mémoire	Variable	Valeur
@1	maPersonne.prenom	
@4	maPersonne.nom	
@11	maPersonne.age	29
@6	**Personne**.isChild()	

EN PRATIQUE

POO	Réalité
Classe	Plans pour fabriquer une personne
Objet (instance de classe)	La personne
Propriétés / attributs	Attributs de la personne (nom, prenom, age, …)
Méthodes / comportement	Fonctionnalités de la personne, ce qu'elle peut faire

Objet	Classe
Objet	Instance de classe
Type d'objet	Le nom de la classe

EN PRATIQUE

Lorsqu'une variable est passée en paramètre d'un sous-programme

- Une copie de la variable passée en paramètre est faite
- Le paramètre est en fait une autre adresse mémoire, mais avec la même valeur
  - Elle a été copiée
  - On dit qu'on transmet par valeur

En objet, la variable passée en paramètre d'un sous-programme n'est plus copiée

- On a accès à la référence mémoire de l'objet
- Chaque modification sur les attributs de cet objet impact donc l'objet qui a été transmit
  - On dit qu'on transmet par référence

EN PRATIQUE

```
Entier a = 5;
Entier b = 6;

a = b;
b = 7;

afficher(a);

//a = 6
```

Variable	Valeur	Adresse mémoire
a	5	@1
b	6	@2
a	6	@1
b	7	@2

```
Personne p1 = nouvelle Personne();
Personne p2 = nouvelle Personne();

p1 = p2;
p2.age = 29;

afficher(p1.age);

//p1.age = 29
```

Variable	Valeur	Adresse mémoire
p1	Personne 1	@1
p2	Personne 2	@2
p1	Personne 2	@2
p2	Personne 2	@2

APPLICATION

Programmation procédurale

- Définition de fonctions, de programmes
- Dirigée par les traitements
- Processus
  - Point d'entrée unique (traitement)
  - En fonction des cas : appelle un traitement ou un autre

Programmation Orientée Objet

- Définition d'objets qui interagissent entre eux
- Dirigée par les données
- Processus
  - Point d'entrée unique (objet)
  - Instancie d'autres objets et les utilise

Meilleure séparation des données et des fonctions qui les manipulent

OBLIGÉ D'INSTANCIER UNE CLASSE ?

Oui, pour manipuler une instance particulière qui a ses propres données

Mais dans certains cas, des programmes peuvent être accessibles sans instanciation

- Les données et/ou les sous-programmes doivent être « Statique »
- En JAVA, c'est le cas de la classe principale qui possède un programme « main » statique !

OBLIGÉ D'INSTANCIER UNE CLASSE ?

```
Classe Personne {
  Caractere[] prenom;
  Caractere[] nom;
  Entier age = 0;

  Méthode Booléen isChild() {
    Si (moi.age < 18) {
      Retour Vrai;
    }
    Retour Faux;
  }

  Méthode Statique Personne creerPersonne(Entier age) {
    Personne maPersonne = nouvelle Personne();
    maPersonne.age = age;
    Retour maPersonne;
  }
}
```

```
Personne pers = Personne.creerPersonne(29);
```

EXERCICE

Créer une classe Joueur

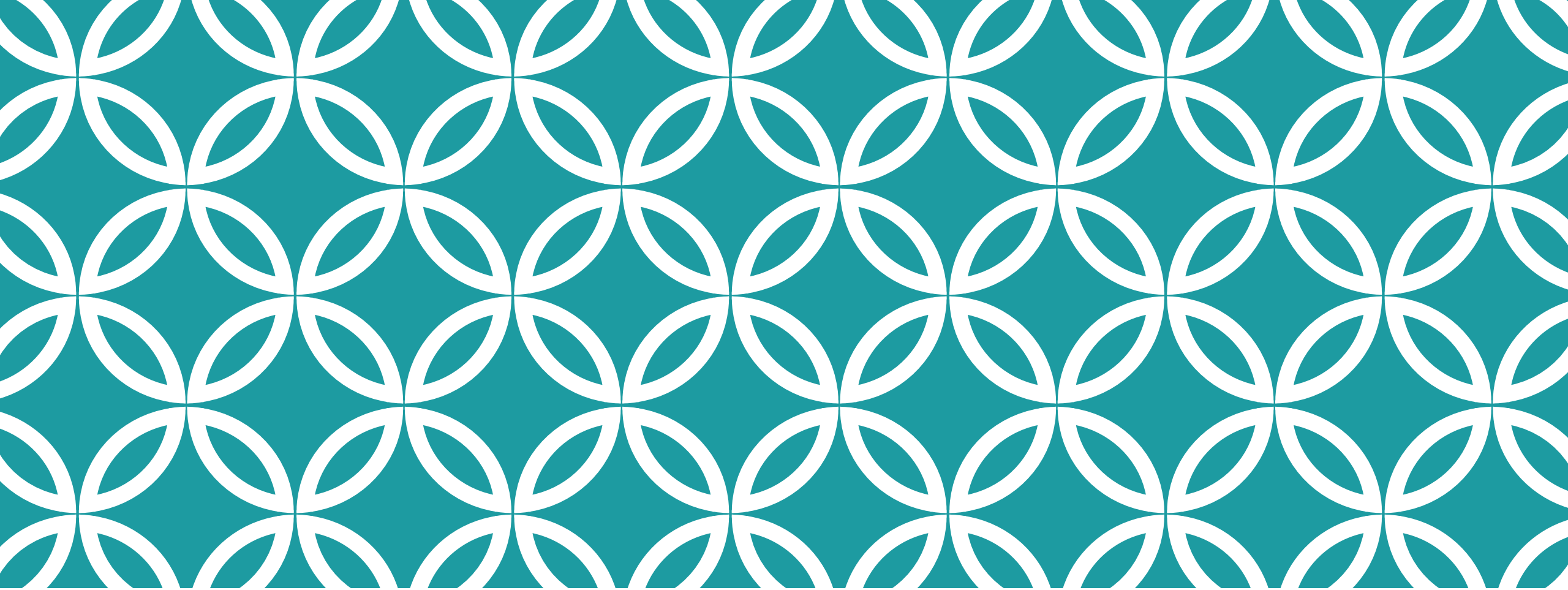
- Nom, Prenom, Age

Créer une classe Equipe

- Nom, 2 joueurs

Dans le programme principal

- Instancier 4 nouveaux joueurs
- Instancier 2 nouvelles équipes
- Affectez les joueurs à une équipe



CES OBJETS BASIQUES

Chaine
Collection

LES CHAINES DE CARACTÈRES

Une chaine de caractères est un tableau de caractères

Introduction d'une classe « Chaine » qui masquera la « complexité » de ce tableau

- C'est un des objectifs de l'objet !

```
Chaine maPhrase = "Une phrase !";
```

Avec la possibilité de concaténer une ou plusieurs chaines

```
Chaine monIntro = "Bonjour, ";
Chaine prenom = "Jérémy";

Chaine maPhrase = monIntro + prenom + " !";
```

Ou de réaliser d'autres opérations, comme comparer 2 chaines

```
Chaine monIntro = "Bonjour, ";
Chaine prenom = "Jérémy";

Si (monIntro.equivaut(prenom)) {
  //...
}
```

EXERCICE

Modifier les classes précédentes

- Remplacer les tableaux de caractères par des Chaine

Ajouter les classes suivantes

- Carte
  - nom, valeur

LES COLLECTIONS

Un gros problème des tableaux, c'est qu'ils ont une taille fixe

- Obligé de créer un nouveau tableau pour l'agrandir
- Obligé de décaler les cases si on veut insérer ou supprimer une valeur
- …

Introduction de la Collection d'objets

```
CollectionDePersonne mesPersonnes = nouvelle CollectionDePersonne();

Personne p1 = nouvelle Personne();
Personne p2 = nouvelle Personne();

mesPersonnes.ajouter(p1);
mesPersonnes.ajouter(p2);

mesPersonnes.obtenir(0);
mesPersonnes.retirer(0);
```

LES COLLECTIONS

Parcourir une collection (boucle Pour)

```
Pour (Entier i = 0; i < mesPersonnes.taille(); i = i + 1) {
  ecrire(mesPersonnes.obtenir(i).nom);
}
```

Parcourir une collection (boucle PourChaque)

```
PourChaque (Personne p dans mesPersonnes) {
  ecrire(p.nom);
}
```

EXERCICE

Dans le programme principal

- Créer une liste de 32 cartes

Afficher le nom de chaque carte