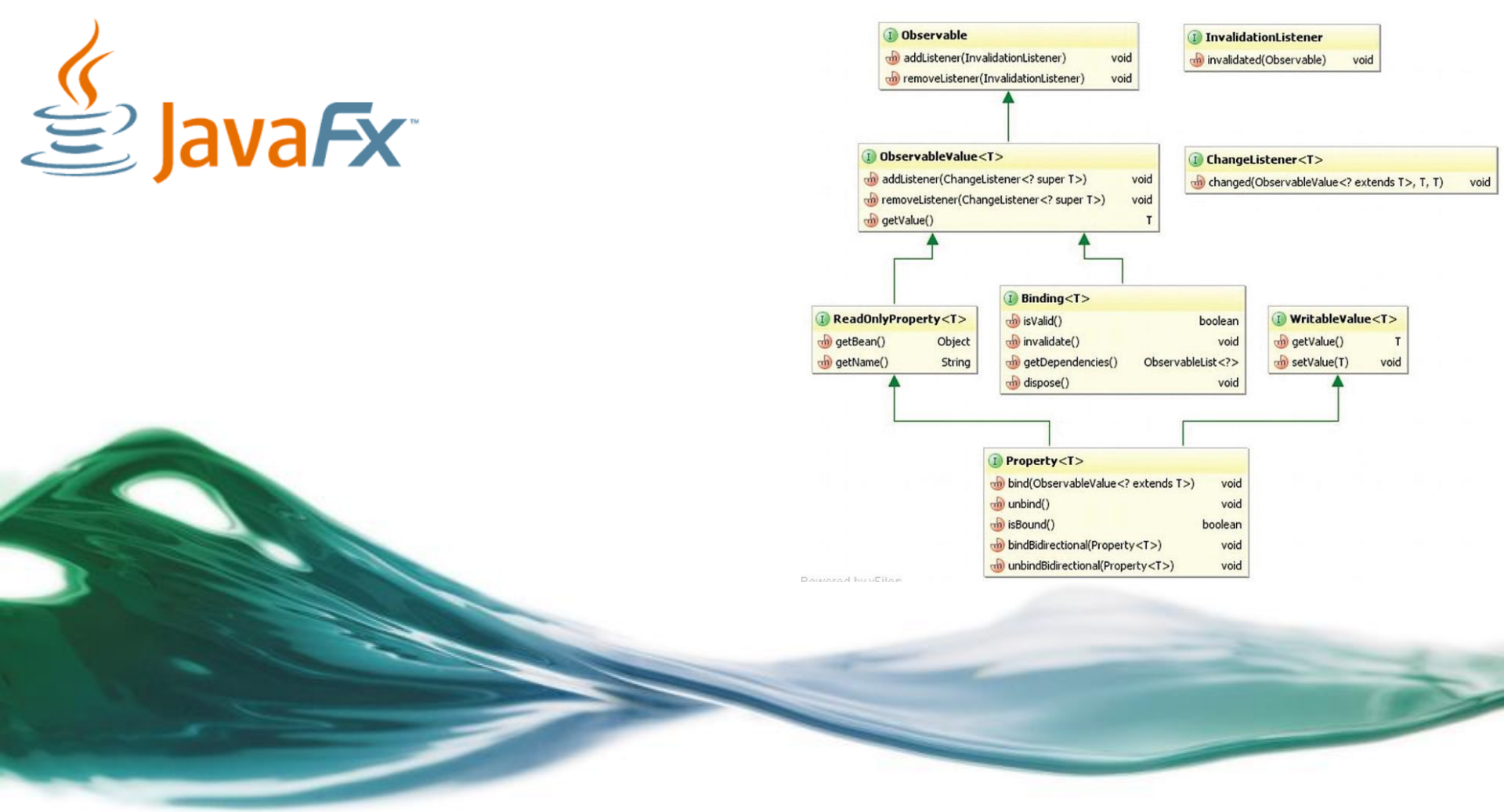


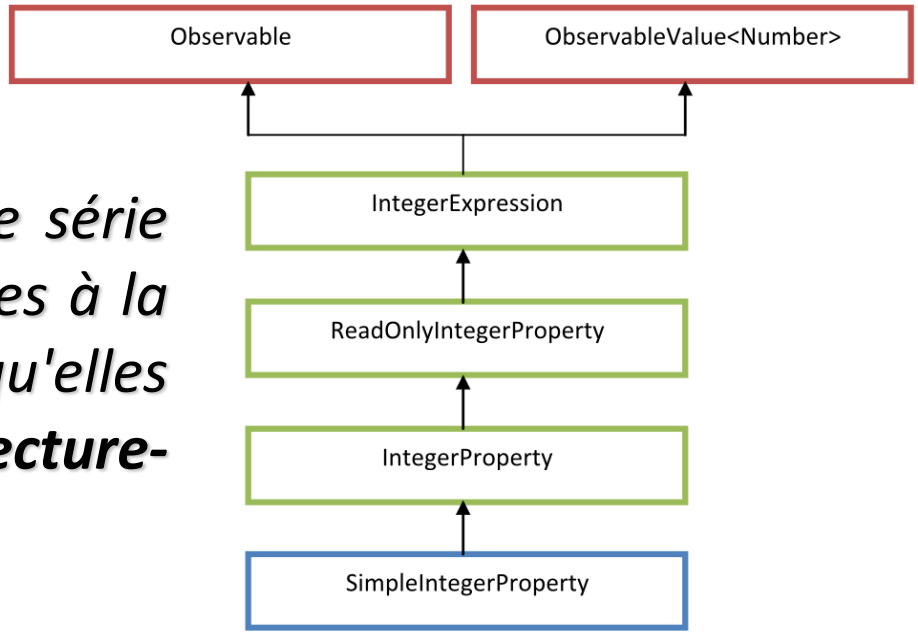
Simple Property Binding

Introduction:

- *JavaFX dispose d'une **API de liaison**, qui permet de **lier une propriété à l'autre**.*

  *Cela signifie que chaque fois que la valeur d'une propriété est **modifiée**, la valeur de la **propriété liée est automatiquement mise à jour**.*

- *JavaFX propose donc toute une série de classes et d'interfaces dédiées à la définition des propriétés qu'elles soient en **lecture seule, en lecture-écriture** .*



Propretés:

- *JavaFX chaque type de de classe à sa propriété:*
  - *SimpleObjectProperty*
  - *SimpleStringProperty*
  - *SimpleIntegerProperty*
  - *…*

Exemple: La classe RectangleProperty

- *Convention d'écriture:*
  - *Utilisez des accesseurs et mutateurs en utilisant les fonctions get et set.*

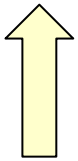
```
public double getHauteur() {
    return hauteur.get();
}
public void setHauteur(double hauteur) {
    this.hauteur.set(hauteur);
}
```

  - *Créer un accesseur pour la **propriété**. Nommez-le en utilisant la convention de nom: **nomDeLaProprieteProperty**.*

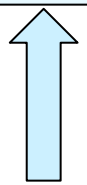
```
public DoubleProperty hauteurProperty() {
    return hauteur;
}
```

Exemple: Binding Simple unidirectionnel

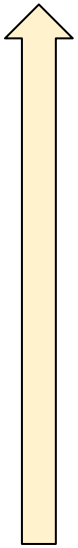
```
perimetre.textProperty().bind(monRectangle.perimetreProperty().asString());
```



le TextField:perimetre est mis à jour



Dés que perimetreProperty de l'objet monRectangle est modifier



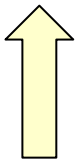
Remarque:

Les deux types de propriétés sont incompatibles entre elles pour le binding.

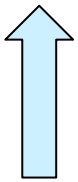
On ne peut pas faire un binding sur une SimpleDoubleProperty et une StringProperty. Double->String la méthode ***asString()***

Exemple: Binding Simple bidirectionnel

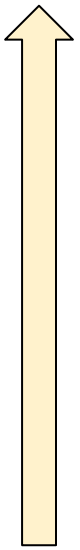
```
hauteur.textProperty().bindBidirectional(monRectangle.hauteurProperty(), sc);
```



le TextField:hauteur est mis à jour



Dés que hauteurProperty de l'objet monRectangle est modifier



Remarque:

Les deux types de propriétés sont incompatibles entre elles pour le binding.

On ne peut pas faire un binding sur une SimpleDoubleProperty et une StringProperty., Mais ici comme le binding est bidirectionnel on utiliser un convertisseur: Double<->String

```
StringConverter sc = new DoubleStringConverter();
```

Exemple: Binding calculé

Ce que l'on veut:

*A partir de deux **DoubleProperty**: hauteur et largeur, calculer avec une mise à jour par binding le périmètre d'un rectangle, qui à son tour sera un **perimetreProperty()** de la classe **rectangleProperty**.*

```
private DoubleProperty hauteur = new SimpleDoubleProperty();
private DoubleProperty largeur = new SimpleDoubleProperty();
private DoubleProperty perimetre = new SimpleDoubleProperty();

...

...

public DoubleProperty perimetreProperty() {
    perimetre.bind((largeur.add(hauteur)).multiply( other: 2));
    return perimetre;
}
```

Exemple: Binding conditionnel

Ce que l'on veut:

SI (le binding calculer surface est supérieur à la constante SEUIL_S)

- *ALORS le binding fixe la couleur du TextField surface au rouge*

- *SINON le binding fixe la couleur du TextField surface au bleu*


```
surface.backgroundProperty().bind(Bindings.when(monRectangle.surfaceProperty().greaterThan(SEUIL_S))
        .then(new Background(new BackgroundFill(Color.RED,  radii: null,  insets: null)))
        .otherwise(new Background(new BackgroundFill(Color.AQUA,  radii: null,  insets: null))));
```