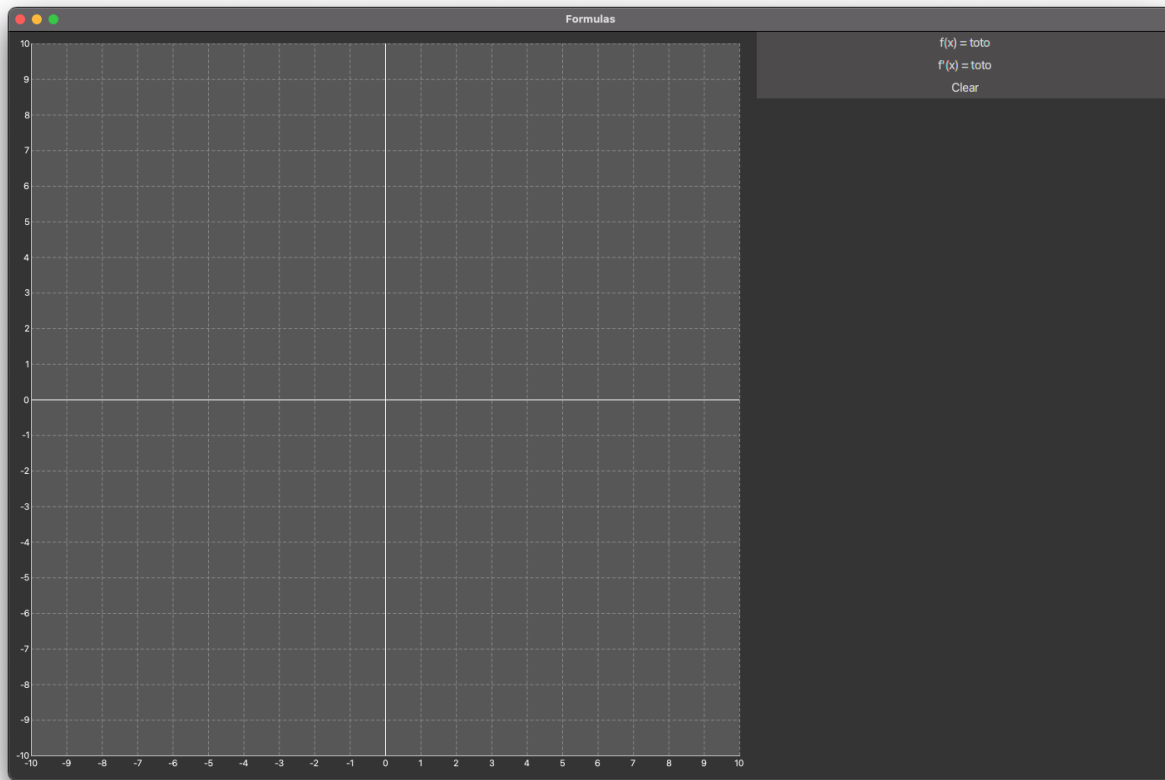


Formules

Consignes pour démarrer le TP

Comme pour les TP précédents, on va utiliser git pour la gestion de versions. Il vous faut donc vous reporter aux consignes du [TP 6](#).

Une fois le dépôt téléchargé, vous pouvez compiler et exécuter le programme en cliquant deux fois sur `formula` -> `application` -> `run`. Vous devriez obtenir l'affichage suivant.



Pour exécuter les tests, il faut passer par l'onglet gradle à droite et cliquer deux fois sur '`color-image`' -> `Tasks` -> `verification` -> `test`. Pour le moment, les tests ne passeront pas car certaines classes sont incomplètes.

Lien vers le projet gitlab à forker pour le TP : [lien projet à forker](#).

Objectif

Dans cette planche de TP, vous allez implémenter des classes pour générer des formules mathématiques. Chaque classe correspondra à un type de formule (constantes, variable, addition, multiplication, ...).

Chaque classe devra implémenter l'interface `Formula` suivante :

```
public interface Formula {

    /**
     * Compute the value of the formula
     *
     * @param xValue the value of the variable x
     * @return the value of the function when the variable x has value {@code xValue}
     */
    double eval(double xValue);

    /**
     * Compute a {@code String} representation of the formula.
     * @return the formula as a {@code String}
     */
    String toString();

    /**
     * Compute the derivative of the formula.
     * @return the derivative of the formula
     */
    Formula derivative();
}
```

Une classe implémentant `Formula` devra donc avoir trois fonctionnalités :

- le calcul de sa valeur étant donnée une valeur pour la variable x : méthode `eval`,
- la représentation en chaîne de caractères de la formule : méthode `toString`,
- le calcul de sa dérivée sous la forme d'une autre formule : méthode `derivative`.

Constante

Le contrat

Vous allez commencer par corriger une classe `Constant` représentant une constante. Cette classe permet de construire une formule correspondant à une constante. Cette classe implémente l'interface `Formula` et contient :

- un constructeur `public Constant(double value)` permettant de créer une constante avec une certaine valeur,
- une méthode `public double eval(double xValue)` qui devra toujours retourner la valeur de la constante,
- une méthode `public String toString()` qui devra retourner la chaîne de caractères correspondant à la constante,
- une méthode `public Formula derivative()` qui devra retourner une formule qui est la dérivée de la constante (indice, pour n'importe quelle fonction $f(x) = c$ avec c constante, la dérivée de la fonction f est la fonction constante définie par $f'(x) = 0$).

Votre implémentation devra passer les tests présents dans la classe `ConstantTest`.

Affichage

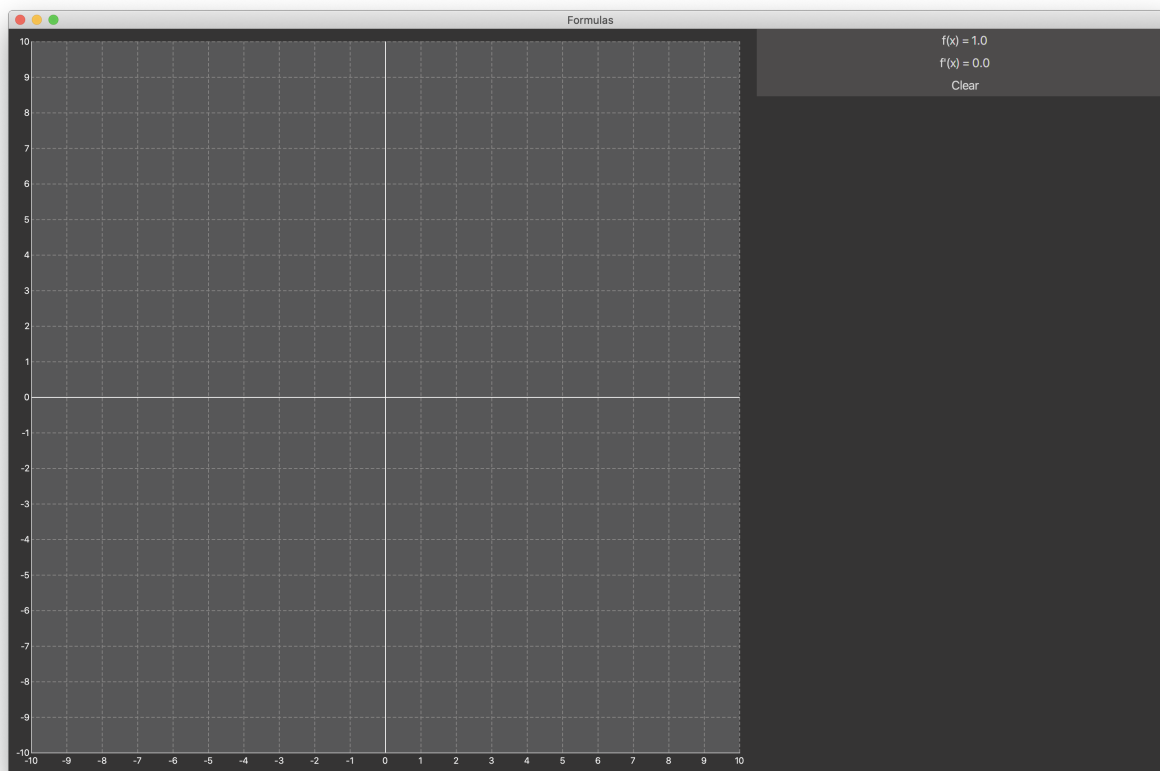
Pour tester votre classe constante, il faut exécuter la tâche `run`. Les fonctions affichées sont définies par les deux lignes suivantes dans le constructeur de la classe `viewer.FunctionList` :

```
PlottableFunction function = new PlottableFunction(new Constant(1), "f");  
addFunctionAndItsDerivative(function);
```

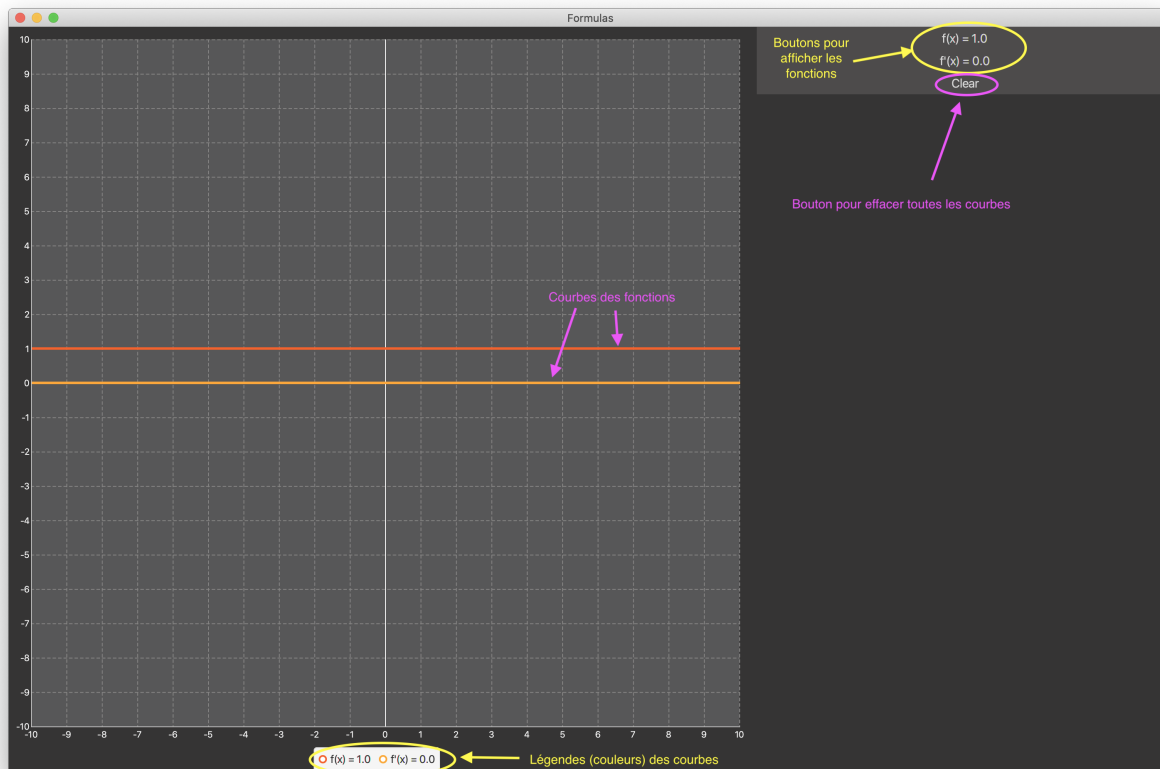
Ces deux lignes permettent respectivement :

- de créer une fonction f dont la formule est une constante égale à 1 et ayant comme nom f ,
- puis de rajouter la fonction f et sa dérivée f' .

Vous devriez obtenir l’affichage suivant.



Pour afficher les fonctions, il suffit de cliquer sur les boutons correspondant. Vous devriez obtenir l’affichage suivant :



Variable x

Le contrat

Vous allez maintenant définir une nouvelle classe **VariableX** représentant une variable x . Cette classe permettra de construire une formule correspondant à la variable x .

Cette classe implémentera l'interface **Formula** et devra contenir :

- un constructeur `public VariableX()` permettant de créer une variable,
- une méthode `public double eval(double xValue)` qui devra toujours retourner la valeur `xValue`,
- une méthode `public String toString()` qui devra retourner la chaîne de caractères correspondant à la variable x et donc la chaîne de caractère "`x`",
- une méthode `public Formula derivative()` qui devra retourner une formule qui est la dérivée en x de la variable x (indice, pour une fonction $g(x) = x$, la dérivée de la fonction g est $g'(x) = 1$).

Le test

Vous pouvez tester votre classe en créant une classe **VariableXTest** s'inspirant de la classe **ConstantTest**.

Affichage

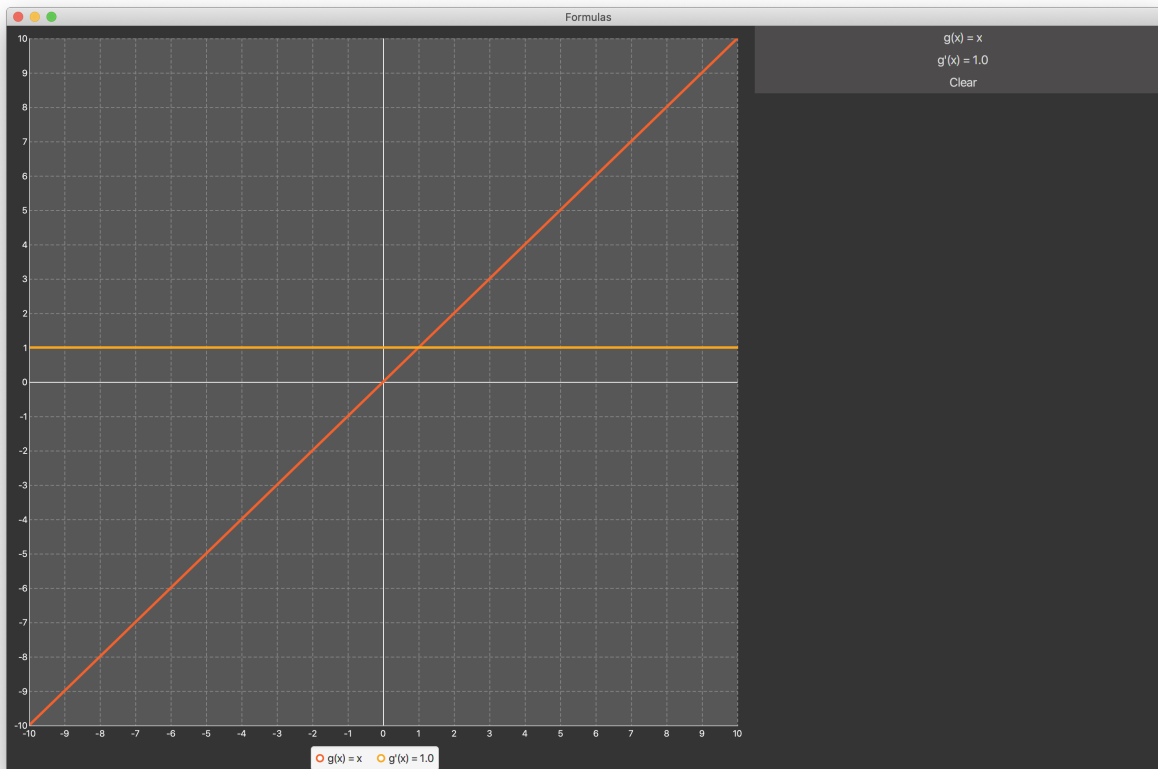
Pour tester votre classe variable, vous allez créer une fonction égale à x dans le logiciel de dessin de fonctions. Pour cela, il vous faut modifier le constructeur de la classe `viewer.FunctionList` pour rajouter les deux lignes suivantes à la place du `//TODO` :

```
PlottableFunction function = new PlottableFunction(new VariableX(), "g");  
addFunctionAndItsDerivative(function);
```

Ces deux lignes permettent respectivement :

- de créer une fonction dont la formule est égale à x et ayant comme nom g ,
- puis de rajouter la fonction g et sa dérivée g' .

Vous devriez obtenir l'affichage suivant.



Addition

Le contrat

Vous allez maintenant définir une classe `Addition` représentant une addition de deux formules. Cette classe permettra de construire une formule correspondant à la somme de 2 formules.

Cette classe implémentera l'interface `Formula` et devra contenir :

- un constructeur `public Addition(Formula leftMember, Formula rightMember)` permettant de créer une addition des deux formules données en arguments correspondants aux deux membres sommés,
- une méthode `public double eval(double xValue)` qui devra toujours retourner la somme des valeurs des deux membres de l'addition,
- une méthode `public String toString()` qui devra retourner la chaîne de caractères correspondant à l'addition, soit la chaîne de caractères correspondant au membre de gauche, concaténée avec le symbole `+` puis concaténée au membre de droite,
- une méthode `public Formula derivative()` qui devra retourner une formule qui est la dérivée en x de la variable x (indice, pour une fonction $h(x) = f(x) + g(x)$, la dérivée de la fonction h est $h'(x) = f'(x) + g'(x)$).

Le test

Vous pouvez tester votre classe en créant une classe `AdditionTest` s'inspirant de la classe `ConstantTest`.

Affichage

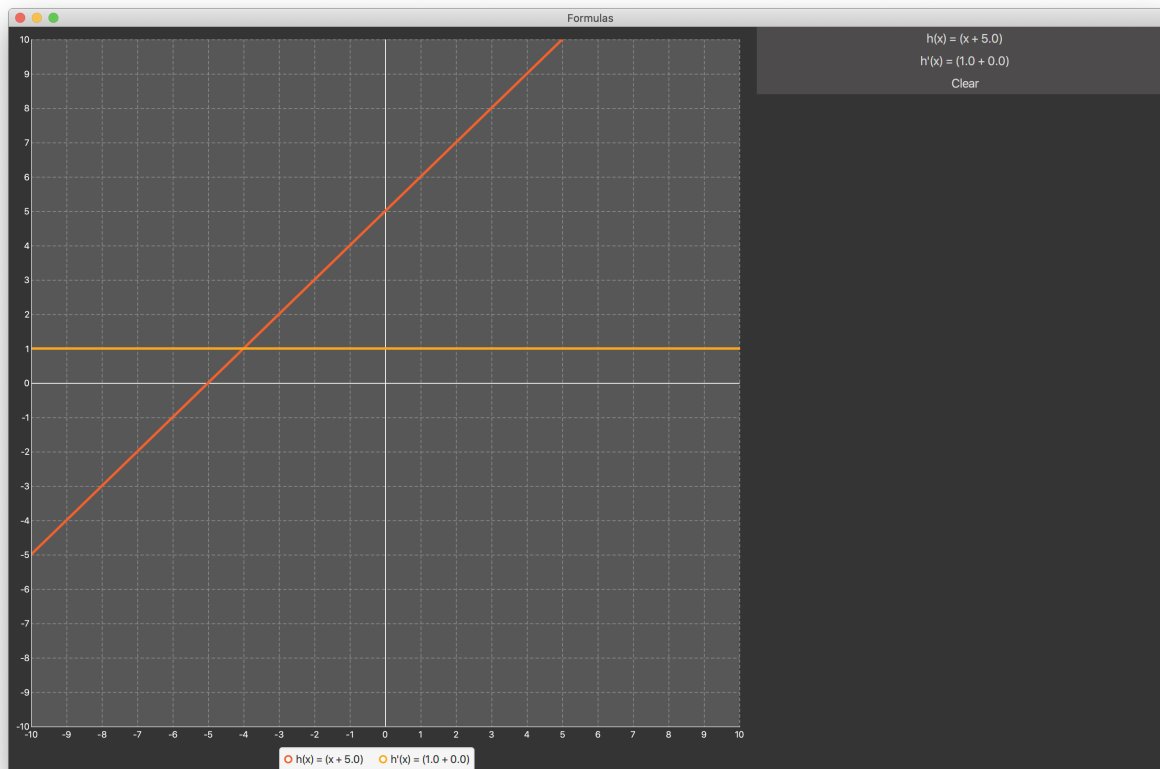
Pour afficher une fonction utilisant la classe `Addition`, vous allez créer une fonction égale à $x + 5$ dans le logiciel de dessin de fonctions. Pour cela, il vous faut modifier le constructeur de la classe `viewer.FunctionList` pour rajouter les deux lignes suivantes à la place du `//TODO` :

```
PlottableFunction function =
    new PlottableFunction(new Addition(new VariableX(),
                                     new Constant(5)), "h");
addFunctionAndItsDerivative(function);
```

Ces deux lignes permettent respectivement :

- de créer une fonction dont la formule est égale à $x + 5$ et ayant comme nom h ,
- puis de rajouter la fonction h et sa dérivée h' .

Vous devriez obtenir l'affichage suivant.



Autres opérations

Écrivez les classes suivantes qui implémentent l'interface **Formule** (n'oubliez pas de les tester) :

- **Multiplication** (multiplication de deux formules),
- **Division** (division d'une formule par une autre),
- **Subtraction** (soustraction d'une formule par une autre),
- **Opposite** (opposé d'une formule),
- **Cosine** (cosinus d'une formule),
- **Sine** (sinus d'une formule),
- **Exponential** (fonction exponentielle d'une formule, c'est-à-dire, e puissance la formule) et
- **Logarithm** (fonction logarithme naturel d'une formule).