

Mandelbrot

1 README	1
1.1 Calcul Distribué de l'Ensemble de Mandelbrot avec Java RMI	1
1.1.1 Aperçu	1
1.1.2 Fonctionnalités	1
1.1.3 Prérequis	1
1.1.4 Démarrage	1
1.2 Distributed Calculation of the Mandelbrot Set with Java RMI	2
1.2.1 Overview	2
1.2.2 Features	2
1.2.3 Prerequisites	2
1.2.4 Getting Started	2
2 README	3
2.1 Calcul Distribué de l'Ensemble de Mandelbrot avec Java RMI	3
2.1.1 Aperçu	3
2.1.2 Fonctionnalités	3
2.1.3 Prérequis	3
2.1.4 Démarrage	3
2.2 Distributed Calculation of the Mandelbrot Set with Java RMI	4
2.2.1 Overview	4
2.2.2 Features	4
2.2.3 Prerequisites	4
2.2.4 Getting Started	4
3 Index des espaces de nommage	5
3.1 Liste des paquetages	5
4 Index hiérarchique	7
4.1 Hiérarchie des classes	7
5 Index des classes	9
5.1 Liste des classes	9
6 Index des fichiers	11
6.1 Liste des fichiers	11
7 Documentation des espaces de nommage	13
7.1 Paquetage client	13
7.2 Paquetage server	13
7.3 Paquetage server.obj	13
8 Documentation des classes	15
8.1 Référence de la classe client.Client	15
8.1.1 Description détaillée	15
8.1.2 Documentation des fonctions membres	15

8.1.2.1 main()	15
8.2 Référence de la classe server.Complexe	16
8.2.1 Documentation des constructeurs et destructeur	16
8.2.1.1 Complexe() [1/2]	16
8.2.1.2 Complexe() [2/2]	16
8.2.2 Documentation des fonctions membres	17
8.2.2.1 add()	17
8.2.2.2 getA()	17
8.2.2.3 getB()	17
8.2.2.4 module()	17
8.2.2.5 multiply()	17
8.2.2.6 setA()	18
8.2.2.7 setB()	18
8.2.2.8 toString()	18
8.3 Référence de la classe server.Constantes	18
8.3.1 Documentation des constructeurs et destructeur	19
8.3.1.1 Constantes()	19
8.3.2 Documentation des fonctions membres	19
8.3.2.1 calculCoordPlan()	20
8.3.2.2 displayInfo()	20
8.3.3 Documentation des données membres	20
8.3.3.1 DECAL_IMAGE_X	20
8.3.3.2 DECAL_IMAGE_Y	20
8.3.3.3 HEIGHT	20
8.3.3.4 HEIGHT_COMPLEXE	20
8.3.3.5 INTERVALLE_FRAME_HEIGHT	21
8.3.3.6 INTERVALLE_FRAME_WIDTH	21
8.3.3.7 LIMIT	21
8.3.3.8 WIDTH	21
8.3.3.9 WIDTH_COMPLEXE	21
8.4 Référence de la classe server.Frame	21
8.4.1 Documentation des constructeurs et destructeur	22
8.4.1.1 Frame()	22
8.4.2 Documentation des fonctions membres	22
8.4.2.1 convert()	22
8.4.2.2 getPanel()	22
8.4.2.3 setStateFrame()	22
8.5 Référence de la classe server.obj.ImpMandelbrot	23
8.5.1 Documentation des constructeurs et destructeur	23
8.5.1.1 ImpMandelbrot()	23
8.5.2 Documentation des fonctions membres	24
8.5.2.1 addResult()	24

8.5.2.2 addTask()	24
8.5.2.3 getMax()	24
8.5.2.4 getTask()	25
8.5.3 Documentation des données membres	25
8.5.3.1 dataToDo	25
8.5.3.2 sizeOfTask	25
8.5.3.3 taskDone	25
8.6 Référence de la classe server.obj.ImpTask	25
8.6.1 Documentation des constructeurs et destructeur	26
8.6.1.1 ImpTask()	26
8.6.2 Documentation des fonctions membres	26
8.6.2.1 convert()	26
8.6.2.2 run()	26
8.7 Référence de l'interface server.obj.Mandelbrot	27
8.7.1 Documentation des fonctions membres	27
8.7.1.1 addResult()	27
8.7.1.2 addTask()	27
8.7.1.3 getTask()	27
8.8 Référence de la classe server.Panel	28
8.8.1 Documentation des constructeurs et destructeur	28
8.8.1.1 Panel()	28
8.8.2 Documentation des fonctions membres	28
8.8.2.1 paintComponent()	28
8.8.2.2 setListePointMandelbrot()	28
8.9 Référence de la classe server.obj.Point	29
8.9.1 Documentation des constructeurs et destructeur	29
8.9.1.1 Point() [1/2]	29
8.9.1.2 Point() [2/2]	29
8.9.2 Documentation des fonctions membres	29
8.9.2.1 getColor()	30
8.9.2.2 getDivergence()	30
8.9.2.3 getX()	30
8.9.2.4 getY()	30
8.9.2.5 setColor()	30
8.9.2.6 setDivergence()	30
8.9.2.7 toString()	30
8.10 Référence de la classe server.Serveur	31
8.10.1 Description détaillée	31
8.10.2 Documentation des fonctions membres	31
8.10.2.1 drawImage()	31
8.10.2.2 main()	31
8.10.3 Documentation des données membres	31

8.10.3.1 numberOfTaskDone	32
8.11 Référence de l'interface server.obj.Task	32
8.11.1 Documentation des fonctions membres	32
8.11.1.1 run()	32
9 Documentation des fichiers	33
9.1 Référence du fichier client/Client.java	33
9.2 Référence du fichier out/production/Mandelbrot/README.md	33
9.3 Référence du fichier README.md	33
9.4 Référence du fichier server/Complexe.java	33
9.5 Référence du fichier server/Constantes.java	34
9.6 Référence du fichier server/Frame.java	34
9.7 Référence du fichier server/obj/ImpMandelbrot.java	34
9.8 Référence du fichier server/obj/ImpTask.java	34
9.9 Référence du fichier server/obj/Mandelbrot.java	35
9.10 Référence du fichier server/obj/Point.java	35
9.11 Référence du fichier server/obj/Task.java	35
9.12 Référence du fichier server/Panel.java	35
9.13 Référence du fichier server/Serveur.java	36

Chapitre 1

README

1.1 Calcul Distribué de l'Ensemble de Mandelbrot avec Java RMI

1.1.1 Aperçu

Ce projet implémente la génération de l'ensemble de Mandelbrot en utilisant une approche de calcul distribué avec Java RMI. L'ensemble de Mandelbrot est un fractal célèbre en mathématiques et cette implémentation permet le calcul parallèle de l'ensemble en utilisant une architecture distribuée.

1.1.2 Fonctionnalités

- **Calcul Distribué** : Utilise Java RMI pour la répartition du calcul sur plusieurs nœuds.
- **Génération de l'Ensemble de Mandelbrot** : Calcule et visualise l'ensemble de Mandelbrot en parallèle.
- **Paramètres Configurables** : Configurez facilement la résolution, le niveau de zoom et d'autres paramètres pour explorer différentes parties de l'ensemble de Mandelbrot.

1.1.3 Prérequis

- Java 8 ou +
- Git

1.1.4 Démarrage

1. **Cloner le Dépôt** :

```
git clone https://github.com/Maxime-Clit/Mandelbrot.git
```

2. **Serveur** :

Il faut d'abord lancer le serveur sur la machine qui va afficher l'ensemble de Mandelbrot.

```
make
```

3. **Client** :

Cette commande va lancer le client qui va se connecter au serveur et lancer les calculs.

```
./exec.sh
```

1.2 Distributed Calculation of the Mandelbrot Set with Java RMI

1.2.1 Overview

This project implements the generation of the Mandelbrot set using a distributed computing approach with Java RMI. The Mandelbrot set is a famous fractal in mathematics, and this implementation allows parallel computation of the set using a distributed architecture.

1.2.2 Features

- **Distributed Calculation:** Uses Java RMI for distributing the computation across multiple nodes.
- **Mandelbrot Set Generation:** Computes and visualizes the Mandelbrot set in parallel.
- **Configurable Parameters:** Easily configure resolution, zoom level, and other parameters to explore different parts of the Mandelbrot set.

1.2.3 Prerequisites

- Java 8 or higher
- Git

1.2.4 Getting Started

1. Clone the Repository:

```
git clone https://github.com/Maxime-Clit/Mandelbrot.git
```

1. Server :

First, launch the server on the machine that will display the Mandelbrot set.

```
make
```

1. Client :

This command will launch the client that connects to the server and initiates the calculations.

```
./exec.sh
```

Chapitre 2

README

2.1 Calcul Distribué de l'Ensemble de Mandelbrot avec Java RMI

2.1.1 Aperçu

Ce projet implémente la génération de l'ensemble de Mandelbrot en utilisant une approche de calcul distribué avec Java RMI. L'ensemble de Mandelbrot est un fractal célèbre en mathématiques et cette implémentation permet le calcul parallèle de l'ensemble en utilisant une architecture distribuée.

2.1.2 Fonctionnalités

- **Calcul Distribué** : Utilise Java RMI pour la répartition du calcul sur plusieurs nœuds.
- **Génération de l'Ensemble de Mandelbrot** : Calcule et visualise l'ensemble de Mandelbrot en parallèle.
- **Paramètres Configurables** : Configurez facilement la résolution, le niveau de zoom et d'autres paramètres pour explorer différentes parties de l'ensemble de Mandelbrot.

2.1.3 Prérequis

- Java 8 ou +
- Git

2.1.4 Démarrage

1. **Cloner le Dépôt** :

```
git clone https://github.com/Maxime-Cllt/Mandelbrot.git
```

2. **Serveur** :

Il faut d'abord lancer le serveur sur la machine qui va afficher l'ensemble de Mandelbrot.

```
make
```

3. **Client** :

Cette commande va lancer le client qui va se connecter au serveur et lancer les calculs.

```
./exec.sh
```

2.2 Distributed Calculation of the Mandelbrot Set with Java RMI

2.2.1 Overview

This project implements the generation of the Mandelbrot set using a distributed computing approach with Java RMI. The Mandelbrot set is a famous fractal in mathematics, and this implementation allows parallel computation of the set using a distributed architecture.

2.2.2 Features

- **Distributed Calculation:** Uses Java RMI for distributing the computation across multiple nodes.
- **Mandelbrot Set Generation:** Computes and visualizes the Mandelbrot set in parallel.
- **Configurable Parameters:** Easily configure resolution, zoom level, and other parameters to explore different parts of the Mandelbrot set.

2.2.3 Prerequisites

- Java 8 or higher
- Git

2.2.4 Getting Started

1. Clone the Repository:

```
git clone https://github.com/Maxime-Clit/Mandelbrot.git
```

1. Server :

First, launch the server on the machine that will display the Mandelbrot set.

```
make
```

1. Client :

This command will launch the client that connects to the server and initiates the calculations.

```
./exec.sh
```

Chapitre 3

Index des espaces de nommage

3.1 Liste des paquetages

Liste des paquetages avec une brève description (si disponible) :

client	13
server	13
server.obj	13

Chapitre 4

Index hiérarchique

4.1 Hiérarchie des classes

Cette liste d'héritage est classée approximativement par ordre alphabétique :

client.Client	15
server.Complexe	16
server.Constantes	18
javax.swing.JFrame	
server.Frame	21
javax.swing.JPanel	
server.Panel	28
server.Serveur	31
Remote	
server.obj.Mandelbrot	27
server.obj.ImpMandelbrot	23
server.obj.Task	32
server.obj.ImpTask	25
Serializable	
server.obj.Point	29
UnicastRemoteObject	
server.obj.ImpMandelbrot	23
server.obj.ImpTask	25

Chapitre 5

Index des classes

5.1 Liste des classes

Liste des classes, structures, unions et interfaces avec une brève description :

client.Client	15
server.Complexe	16
server.Constantes	18
server.Frame	21
server.obj.ImpMandelbrot	23
server.obj.ImpTask	25
server.obj.Mandelbrot	27
server.Panel	28
server.obj.Point	29
server.Serveur	31
server.obj.Task	32

Chapitre 6

Index des fichiers

6.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

client/ Client.java	33
server/ Complexe.java	33
server/ Constantes.java	34
server/ Frame.java	34
server/ Panel.java	35
server/ Serveur.java	36
server/obj/ ImpMandelbrot.java	34
server/obj/ ImpTask.java	34
server/obj/ Mandelbrot.java	35
server/obj/ Point.java	35
server/obj/ Task.java	35

Chapitre 7

Documentation des espaces de nommage

7.1 Paquetage client

Classes

— class [Client](#)

7.2 Paquetage server

Paquetages

— package [obj](#)

Classes

— class [Complexe](#)
— class [Constantes](#)
— class [Frame](#)
— class [Panel](#)
— class [Serveur](#)

7.3 Paquetage server.obj

Classes

— class [ImpMandelbrot](#)

- class [ImpTask](#)
 - interface [Mandelbrot](#)
 - class [Point](#)
 - interface [Task](#)
-

Chapitre 8

Documentation des classes

8.1 Référence de la classe client.Client

Fonctions membres publiques statiques

— static void `main` (String[] args)

8.1.1 Description détaillée

Classe Client qui va se connecter au serveur et récupérer les données à traiter

args[0] : ip du serveur

ex: java `client.Client` localhost

8.1.2 Documentation des fonctions membres

8.1.2.1 `main()`

```
static void client.Client.main (  
    String[] args ) [static]
```

Voici le graphe d'appel pour cette fonction :

8.2 Référence de la classe server.Complexe

Fonctions membres publiques

- [Complexe](#) ()
- [Complexe](#) (double newA, double newB)
- [Complexe multiply](#) ([Complexe](#) complexe)
- [Complexe add](#) ([Complexe](#) complexe)
- double [module](#) ()
- double [getA](#) ()
- void [setA](#) (double newA)
- double [getB](#) ()
- void [setB](#) (double newB)
- String [toString](#) ()

8.2.1 Documentation des constructeurs et destructeur

8.2.1.1 [Complexe\(\)](#) [1/2]

```
server.Complexe.Complexe ( )
```

Constructeur de la classe Complexe qui initialise les deux doubles à 0 Voici le graphe des appelants de cette fonction :

8.2.1.2 [Complexe\(\)](#) [2/2]

```
server.Complexe.Complexe (
    double newA,
    double newB )
```

Constructeur de la classe Complexe qui prend en paramètre deux doubles

Paramètres

<i>newA</i>	(double) les deux doubles qui vont initialiser les deux doubles de la classe
<i>newB</i>	(double) les deux doubles qui vont initialiser les deux doubles de la classe

8.2.2 Documentation des fonctions membres

8.2.2.1 add()

```
Complexe server.Complexe.add (
    Complexe complexe )
```

Méthode qui calcule la somme de deux complexes

Paramètres

<i>complexe</i>	
-----------------	--

Renvoie

un nouveau complexe qui est la somme des deux complexes (Complexe)

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

8.2.2.2 getA()

```
double server.Complexe.getA ( )
```

Voici le graphe des appelants de cette fonction :

8.2.2.3 getB()

```
double server.Complexe.getB ( )
```

Voici le graphe des appelants de cette fonction :

8.2.2.4 module()

```
double server.Complexe.module ( )
```

Méthode qui calcule la différence de deux complexes

Renvoie

le module du complexe en double arrondi à 2 chiffres après la virgule (double)

Voici le graphe des appelants de cette fonction :

8.2.2.5 multiply()

```
Complexe server.Complexe.multiply (
    Complexe complexe )
```

Méthode qui calcule le produit de deux complexes $(a+ib)*(c+id) = (ac-bd)+i(ad+bc)$ et qui retourne un nouveau complexe

Paramètres

<i>complexe</i>	(Complexe) le complexe avec lequel on veut multiplier
-----------------	---

Renvoie

un nouveau complexe qui est le produit des deux complexes (Complexe)

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

8.2.2.6 setA()

```
void server.Complexe.setA (  
    double newA )
```

8.2.2.7 setB()

```
void server.Complexe.setB (  
    double newB )
```

8.2.2.8 toString()

```
String server.Complexe.toString ( )
```

Voici le graphe d'appel pour cette fonction :

La documentation de cette classe a été générée à partir du fichier suivant :

— [server/Complexe.java](#)

8.3 Référence de la classe server.Constantes

Graphe de collaboration de server.Constantes:

Fonctions membres publiques

— [Constantes](#) (int width, int height, [Complexe](#) widthComplexe, [Complexe](#) heightComplexe, int limit)

Fonctions membres publiques statiques

— static void [calculCoordPlan](#) ()

— static void [displayInfo](#) ()

Attributs publics statiques

- static int `WIDTH` = 500
- static int `HEIGHT` = 250
- static `Complexe` `WIDTH_COMPLEXE` = new `Complexe`(-2, 1)
- static `Complexe` `HEIGHT_COMPLEXE` = new `Complexe`(1, -1)
- static int `LIMIT` = 200
- static double `INTERVALLE_FRAME_WIDTH` = Math.abs(WIDTH_COMPLEXE.getA() - HEIGHT_COMPLEXE.getA())
- static double `INTERVALLE_FRAME_HEIGHT` = Math.abs(WIDTH_COMPLEXE.getB() - HEIGHT_COMPLEXE.getB())
- static double `DECAL_IMAGE_X` = Math.min(WIDTH_COMPLEXE.getA(), HEIGHT_COMPLEXE.getA())
- static double `DECAL_IMAGE_Y` = Math.min(WIDTH_COMPLEXE.getB(), HEIGHT_COMPLEXE.getB())

8.3.1 Documentation des constructeurs et destructeur

8.3.1.1 Constantes()

```
server.Constantes.Constantes (
    int width,
    int height,
    Complexe widthComplexe,
    Complexe heightComplexe,
    int limit )
```

Constructeur de la classe Constantes qui initialise les constantes de l'application

Paramètres

<i>width</i>	Largeur de la frame
<i>height</i>	Hauteur de la frame
<i>widthComplexe</i>	Intervalle de calcul en x
<i>heightComplexe</i>	Intervalle de calcul en y
<i>limit</i>	Limite de calcul du module de mandelbrot

8.3.2 Documentation des fonctions membres

8.3.2.1 calculCoordPlan()

```
static void server.Constantes.calculCoordPlan ( ) [static]
```

Calcul les coordonnées de l'image dans le plan complexe en fonction de l'intervalle de calcul Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

8.3.2.2 displayInfo()

```
static void server.Constantes.displayInfo ( ) [static]
```

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

8.3.3 Documentation des données membres

8.3.3.1 DECAL_IMAGE_X

```
double server.Constantes.DECAL_IMAGE_X = Math.min(WIDTH_COMPLEXE.getA(), HEIGHT_COMPLEXE.↵  
getA()) [static]
```

8.3.3.2 DECAL_IMAGE_Y

```
double server.Constantes.DECAL_IMAGE_Y = Math.min(WIDTH_COMPLEXE.getB(), HEIGHT_COMPLEXE.↵  
getB()) [static]
```

8.3.3.3 HEIGHT

```
int server.Constantes.HEIGHT = 250 [static]
```

8.3.3.4 HEIGHT_COMPLEXE

```
Complexe server.Constantes.HEIGHT_COMPLEXE = new Complexe(1, -1) [static]
```

8.3.3.5 INTERVALLE_FRAME_HEIGHT

```
double server.Constantes.INTERVALLE_FRAME_HEIGHT = Math.abs(WIDTH_COMPLEXE.getB() - HEIGHT_COMPLEXE.getB()) [static]
```

8.3.3.6 INTERVALLE_FRAME_WIDTH

```
double server.Constantes.INTERVALLE_FRAME_WIDTH = Math.abs(WIDTH_COMPLEXE.getA() - HEIGHT_COMPLEXE.getA()) [static]
```

8.3.3.7 LIMIT

```
int server.Constantes.LIMIT = 200 [static]
```

8.3.3.8 WIDTH

```
int server.Constantes.WIDTH = 500 [static]
```

8.3.3.9 WIDTH_COMPLEXE

```
Complexe server.Constantes.WIDTH_COMPLEXE = new Complexe(-2, 1) [static]
```

La documentation de cette classe a été générée à partir du fichier suivant :

— [server/Constantes.java](#)

8.4 Référence de la classe server.Frame

Graphe d'héritage de server.Frame:

Graphe de collaboration de server.Frame:

Fonctions membres publiques

- [Frame](#) (int width, int height)
 - [Panel](#) [getPanel](#) ()
 - void [setStateFrame](#) (boolean state)
 - final [Complexe](#) [convert](#) (final [Point](#) p)
-

8.4.1 Documentation des constructeurs et destructeur

8.4.1.1 Frame()

```
server.Frame.Frame (
    int width,
    int height )
```

8.4.2 Documentation des fonctions membres

8.4.2.1 convert()

```
final Complexe server.Frame.convert (
    final Point p )
```

Convertit un point en son équivalent complexe dans l'intervalle donné

Paramètres

<i>p</i>	Point à convertir
----------	-------------------

Renvoie

Complexe correspondant au point

Voici le graphe d'appel pour cette fonction :

8.4.2.2 getPanel()

```
Panel server.Frame.getPanel ( )
```

Voici le graphe des appelants de cette fonction :

8.4.2.3 setStateFrame()

```
void server.Frame.setStateFrame (
    boolean state )
```

Voici le graphe des appelants de cette fonction :

La documentation de cette classe a été générée à partir du fichier suivant :

— [server/Frame.java](#)

8.5 Référence de la classe server.obj.ImpMandelbrot

Graphe d'héritage de server.obj.ImpMandelbrot:

Graphe de collaboration de server.obj.ImpMandelbrot:

Fonctions membres publiques

- `ImpMandelbrot ()` throws `RemoteException`
- void `addTask` (final `Point` point) throws `RemoteException`
- short `getMax` () throws `RemoteException`
- `Task` `getTask` () throws `RemoteException`
- void `addResult` (final `Task` task) throws `RemoteException`
- void `addTask` (final `Point` p) throws `RemoteException`
- void `addResult` (final `Task` t) throws `RemoteException`
- `Task` `getTask` () throws `RemoteException`

Attributs publics

- List< `Point` > `dataToDo`
- List< `Task` > `taskDone`
- int `sizeOfTask`

8.5.1 Documentation des constructeurs et destructeur

8.5.1.1 ImpMandelbrot()

```
server.obj.ImpMandelbrot.ImpMandelbrot ( ) throws RemoteException
```

Constructeur de la classe ImpMandelbrot

Exceptions

<i>RemoteException</i>	
------------------------	--

8.5.2 Documentation des fonctions membres

8.5.2.1 addResult()

```
void server.obj.ImpMandelbrot.addResult (
    final Task task ) throws RemoteException
```

Fonction qui retourne le nombre de tâches à traiter

Paramètres

<i>task</i>	Task à ajouter à la liste des tâches traitées
-------------	---

Exceptions

<i>RemoteException</i>	
------------------------	--

Implémente [server.obj.Mandelbrot](#).

8.5.2.2 addTask()

```
void server.obj.ImpMandelbrot.addTask (
    final Point point ) throws RemoteException
```

Fonction qui ajoute une tâche à traiter

Paramètres

<i>point</i>	Point à ajouter à la liste des tâches à traiter (dataToDo)
--------------	--

Exceptions

<i>RemoteException</i>	
------------------------	--

Implémente [server.obj.Mandelbrot](#).

8.5.2.3 getMax()

```
short server.obj.ImpMandelbrot.getMax ( ) throws RemoteException
```

Voici le graphe des appelants de cette fonction :

8.5.2.4 getTask()

```
Task server.obj.ImpMandelbrot.getTask ( ) throws RemoteException
```

Fonction qui retourne une tâche à traiter

Renvoie

Task

Exceptions

<i>RemoteException</i>	
------------------------	--

Implémente [server.obj.Mandelbrot](#).

8.5.3 Documentation des données membres

8.5.3.1 dataToDo

```
List<Point> server.obj.ImpMandelbrot.dataToDo
```

8.5.3.2 sizeOfTask

```
int server.obj.ImpMandelbrot.sizeOfTask
```

8.5.3.3 taskDone

```
List<Task> server.obj.ImpMandelbrot.taskDone
```

La documentation de cette classe a été générée à partir du fichier suivant :

— [server/obj/ImpMandelbrot.java](#)

8.6 Référence de la classe server.obj.ImpTask

Graphe d'héritage de server.obj.ImpTask:

Graphe de collaboration de server.obj.ImpTask:

Fonctions membres publiques

- [ImpTask](#) (final [Point](#) point) throws RemoteException
- void [run](#) () throws RemoteException
- [Complexe convert](#) (final [Point](#) p)
- void [run](#) () throws RemoteException

8.6.1 Documentation des constructeurs et destructeur

8.6.1.1 ImpTask()

```
server.obj.ImpTask.ImpTask (
    final Point point ) throws RemoteException
```

8.6.2 Documentation des fonctions membres

8.6.2.1 convert()

```
Complexe server.obj.ImpTask.convert (
    final Point p )
```

Convertit un point en complexe pour le calcul de la fractale

Paramètres

<i>p</i>	Point à convertir
----------	-------------------

Renvoie

Complexe

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

8.6.2.2 run()

```
void server.obj.ImpTask.run ( ) throws RemoteException
```

Implémente [server.obj.Task](#).

Voici le graphe d'appel pour cette fonction :

La documentation de cette classe a été générée à partir du fichier suivant :

- [server/obj/ImpTask.java](#)

8.7 Référence de l'interface `server.obj.Mandelbrot`

Graphe d'héritage de `server.obj.Mandelbrot`:

Graphe de collaboration de `server.obj.Mandelbrot`:

Fonctions membres publiques

- void `addTask` (final `Point` p) throws `RemoteException`
- void `addResult` (final `Task` t) throws `RemoteException`
- `Task` `getTask` () throws `RemoteException`

8.7.1 Documentation des fonctions membres

8.7.1.1 `addResult()`

```
void server.obj.Mandelbrot.addResult (  
    final Task t ) throws RemoteException
```

Implémenté dans `server.obj.ImpMandelbrot`.

Voici le graphe des appelants de cette fonction :

8.7.1.2 `addTask()`

```
void server.obj.Mandelbrot.addTask (  
    final Point p ) throws RemoteException
```

Implémenté dans `server.obj.ImpMandelbrot`.

8.7.1.3 `getTask()`

```
Task server.obj.Mandelbrot.getTask ( ) throws RemoteException
```

Implémenté dans `server.obj.ImpMandelbrot`.

Voici le graphe des appelants de cette fonction :

La documentation de cette interface a été générée à partir du fichier suivant :

- `server/obj/Mandelbrot.java`
-

8.8 Référence de la classe server.Panel

Graphe d'héritage de server.Panel:

Graphe de collaboration de server.Panel:

Fonctions membres publiques

- [Panel](#) ()
- void [setListePointMandelbrot](#) (List< [Point](#) > listePointMandelbrot)

Fonctions membres protégées

- void [paintComponent](#) (Graphics g)

8.8.1 Documentation des constructeurs et destructeur

8.8.1.1 Panel()

```
server.Panel.Panel ( )
```

8.8.2 Documentation des fonctions membres

8.8.2.1 paintComponent()

```
void server.Panel.paintComponent (
    Graphics g ) [protected]
```

8.8.2.2 setListePointMandelbrot()

```
void server.Panel.setListePointMandelbrot (
    List< Point > listePointMandelbrot )
```

Voici le graphe des appelants de cette fonction :

La documentation de cette classe a été générée à partir du fichier suivant :

- [server/Panel.java](#)
-

8.9 Référence de la classe server.obj.Point

Graphe d'héritage de server.obj.Point:

Graphe de collaboration de server.obj.Point:

Fonctions membres publiques

- [Point](#) (final int x, final int y)
- [Point](#) (final int x, final int y, final Color color)
- int [getX](#) ()
- int [getY](#) ()
- int [getDivergence](#) ()
- void [setDivergence](#) (final short newDivergence)
- Color [getColor](#) ()
- void [setColor](#) (final Color newcolor)
- String [toString](#) ()

8.9.1 Documentation des constructeurs et destructeur

8.9.1.1 [Point\(\)](#) [1/2]

```
server.obj.Point.Point (
    final int x,
    final int y )
```

8.9.1.2 [Point\(\)](#) [2/2]

```
server.obj.Point.Point (
    final int x,
    final int y,
    final Color color )
```

8.9.2 Documentation des fonctions membres

8.9.2.1 getColor()

```
Color server.obj.Point.getColor ( )
```

8.9.2.2 getDivergence()

```
int server.obj.Point.getDivergence ( )
```

8.9.2.3 getX()

```
int server.obj.Point.getX ( )
```

Voici le graphe des appelants de cette fonction :

8.9.2.4 getY()

```
int server.obj.Point.getY ( )
```

Voici le graphe des appelants de cette fonction :

8.9.2.5 setColor()

```
void server.obj.Point.setColor (
    final Color newcolor )
```

Voici le graphe des appelants de cette fonction :

8.9.2.6 setDivergence()

```
void server.obj.Point.setDivergence (
    final short newDivergence )
```

Voici le graphe des appelants de cette fonction :

8.9.2.7 toString()

```
String server.obj.Point.toString ( )
```

La documentation de cette classe a été générée à partir du fichier suivant :

— [server/obj/Point.java](#)

8.10 Référence de la classe server.Serveur

Fonctions membres publiques statiques

- static void `main` (String[] args) throws RemoteException
- static void `drawImage` ()

Attributs publics statiques

- static long `numberOfTaskDone` = 0

8.10.1 Description détaillée

Classe Serveur args[0] : largeur de l'image args[1] : hauteur de l'image args[2] : nombre de limite de divergence args[3] : partie réelle de la borne inférieure du plan complexe args[4] : partie imaginaire de la borne inférieure du plan complexe args[5] : partie réelle de la borne supérieure du plan complexe args[6] : partie imaginaire de la borne supérieure du plan complexe

exemple : java Serveur 800 600 1000 -1.5 -1.5 1.5 1.5

8.10.2 Documentation des fonctions membres

8.10.2.1 drawImage()

```
static void server.Serveur.drawImage ( ) [static]
```

Méthode drawImage qui permet de dessiner l'image de la fractale de Mandelbrot en fonction des points de la liste. Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

8.10.2.2 main()

```
static void server.Serveur.main (
    String[] args ) throws RemoteException [static]
```

Voici le graphe d'appel pour cette fonction :

8.10.3 Documentation des données membres

8.10.3.1 numberOfTaskDone

```
long server.Serveur.numberOfTaskDone = 0 [static]
```

La documentation de cette classe a été générée à partir du fichier suivant :

— [server/Serveur.java](#)

8.11 Référence de l'interface server.obj.Task

Graphe d'héritage de server.obj.Task:

Graphe de collaboration de server.obj.Task:

Fonctions membres publiques

— void [run](#) () throws RemoteException

8.11.1 Documentation des fonctions membres

8.11.1.1 run()

```
void server.obj.Task.run ( ) throws RemoteException
```

Implémenté dans [server.obj.ImpTask](#).

Voici le graphe des appelants de cette fonction :

La documentation de cette interface a été générée à partir du fichier suivant :

— [server/obj/Task.java](#)

Chapitre 9

Documentation des fichiers

9.1 Référence du fichier client/Client.java

Classes

— class [client.Client](#)

Paquetages

— package [client](#)

9.2 Référence du fichier out/production/Mandelbrot/README.md

9.3 Référence du fichier README.md

9.4 Référence du fichier server/Complexe.java

Classes

— class [server.Complexe](#)

Paquetages

— package [server](#)

9.5 Référence du fichier server/Constantes.java

Classes

— class [server.Constantes](#)

Paquetages

— package [server](#)

9.6 Référence du fichier server/Frame.java

Classes

— class [server.Frame](#)

Paquetages

— package [server](#)

9.7 Référence du fichier server/obj/ImpMandelbrot.java

Classes

— class [server.obj.ImpMandelbrot](#)

Paquetages

— package [server.obj](#)

9.8 Référence du fichier server/obj/ImpTask.java

Classes

— class [server.obj.ImpTask](#)

Paquetages

— package [server.obj](#)

9.9 Référence du fichier server/obj/Mandelbrot.java

Classes

— interface [server.obj.Mandelbrot](#)

Paquetages

— package [server.obj](#)

9.10 Référence du fichier server/obj/Point.java

Classes

— class [server.obj.Point](#)

Paquetages

— package [server.obj](#)

9.11 Référence du fichier server/obj/Task.java

Classes

— interface [server.obj.Task](#)

Paquetages

— package [server.obj](#)

9.12 Référence du fichier server/Panel.java

Classes

— class [server.Panel](#)

Paquetages

— package [server](#)

9.13 Référence du fichier server/Serveur.java

Classes

— class [server.Serveur](#)

Paquetages

— package [server](#)
