

labo\_08\_gachet\_jean\_gallay\_david

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>File Index</b>	<b>1</b>
1.1	File List . . . . .	1
<b>2</b>	<b>File Documentation</b>	<b>3</b>
2.1	game_of_life.h File Reference . . . . .	3
2.1.1	Macro Definition Documentation . . . . .	3
2.1.1.1	ALIVE . . . . .	3
2.1.1.2	ALIVE_CHAR . . . . .	3
2.1.1.3	DEAD . . . . .	4
2.1.1.4	DEAD_CHAR . . . . .	4
2.1.1.5	HEIGHT . . . . .	4
2.1.1.6	WIDTH . . . . .	4
2.1.2	Function Documentation . . . . .	4
2.1.2.1	computeMultipleGens() . . . . .	4
2.1.2.2	computeNextGen() . . . . .	5
2.1.2.3	displayGame() . . . . .	5
2.1.2.4	preFillGame() . . . . .	5
	<b>Index</b>	<b>7</b>



# Chapter 1

## File Index

### 1.1 File List

Here is a list of all files with brief descriptions:

<a href="#">game_of_life.h</a> . . . . .	3
--	---



## Chapter 2

# File Documentation

### 2.1 game\_of\_life.h File Reference

#### Macros

- `#define HEIGHT 9`
- `#define WIDTH 9`
- `#define ALIVE true`
- `#define DEAD false`
- `#define ALIVE_CHAR 'X'`
- `#define DEAD_CHAR 'O'`

#### Functions

- `bool computeNextGen (bool currentGen[HEIGHT][WIDTH])`
- `void computeMultipleGens (bool currentGen[HEIGHT][WIDTH], unsigned n, bool autoStop=true)`
- `void displayGame (const bool game[HEIGHT][WIDTH])`
- `void preFillGame (bool game[HEIGHT][WIDTH])`

#### 2.1.1 Macro Definition Documentation

##### 2.1.1.1 ALIVE

```
#define ALIVE true
```

##### 2.1.1.2 ALIVE\_CHAR

```
#define ALIVE_CHAR 'X'
```

### 2.1.1.3 DEAD

```
#define DEAD false
```

### 2.1.1.4 DEAD\_CHAR

```
#define DEAD_CHAR 'O'
```

### 2.1.1.5 HEIGHT

```
#define HEIGHT 9
```

### 2.1.1.6 WIDTH

```
#define WIDTH 9
```

## 2.1.2 Function Documentation

### 2.1.2.1 computeMultipleGens()

```
void computeMultipleGens (
    bool currentGen[HEIGHT][WIDTH],
    unsigned n,
    bool autoStop = true )
```

Take a game and computes the *n* next generations (stops if a stable state is reached). Displays the game after each generation.

#### Parameters

<i>currentGen</i>	[IN] current state of the game [OUT] new state of the game
<i>n</i>	number of generations to compute
<i>autoStop</i>	stops if no more changes are detected between generations



### 2.1.2.2 computeNextGen()

```
bool computeNextGen (
    bool currentGen[HEIGHT][WIDTH] )
```

Take a game and computes the next generation.

#### Parameters

<i>currentGen</i>	[IN] current state of the game [OUT] new state of the game
-------------------	--

#### Returns

if a change occurred

### 2.1.2.3 displayGame()

```
void displayGame (
    const bool game[HEIGHT][WIDTH] )
```

Display the game.

#### Parameters

<i>game</i>	
-------------	--

### 2.1.2.4 preFillGame()

```
void preFillGame (
    bool game[HEIGHT][WIDTH] )
```

Prefills the game with dead cells.

#### Parameters

<i>game</i>	Bi-dimensional bool array
-------------	---------------------------



# Index

ALIVE\_CHAR  
game\_of\_life.h, 3

ALIVE  
game\_of\_life.h, 3

computeMultipleGens  
game\_of\_life.h, 4

computeNextGen  
game\_of\_life.h, 4

DEAD\_CHAR  
game\_of\_life.h, 4

DEAD  
game\_of\_life.h, 3

displayGame  
game\_of\_life.h, 5

game\_of\_life.h, 3  
ALIVE\_CHAR, 3  
ALIVE, 3  
computeMultipleGens, 4  
computeNextGen, 4  
DEAD\_CHAR, 4  
DEAD, 3  
displayGame, 5  
HEIGHT, 4  
preFillGame, 5  
WIDTH, 4

HEIGHT  
game\_of\_life.h, 4

preFillGame  
game\_of\_life.h, 5

WIDTH  
game\_of\_life.h, 4