

# Arcade

1.0

Generated by Doxygen 1.8.15



<b>1 How to implement your own game / library</b>	<b>1</b>
<b>2 Namespace Index</b>	<b>3</b>
2.1 Namespace List . . . . .	3
<b>3 Hierarchical Index</b>	<b>5</b>
3.1 Class Hierarchy . . . . .	5
<b>4 Class Index</b>	<b>7</b>
4.1 Class List . . . . .	7
<b>5 Namespace Documentation</b>	<b>9</b>
5.1 Arcade Namespace Reference . . . . .	9
5.1.1 Detailed Description . . . . .	9
5.2 Arcade::Display Namespace Reference . . . . .	9
5.2.1 Detailed Description . . . . .	9
5.3 Arcade::Exceptions Namespace Reference . . . . .	10
5.3.1 Detailed Description . . . . .	10
5.4 Arcade::Games Namespace Reference . . . . .	10
5.4.1 Detailed Description . . . . .	10
<b>6 Class Documentation</b>	<b>11</b>
6.1 Arcade::Games::AGameModule Class Reference . . . . .	11
6.1.1 Detailed Description . . . . .	12
6.1.2 Constructor & Destructor Documentation . . . . .	12
6.1.2.1 AGameModule() . . . . .	12
6.1.3 Member Function Documentation . . . . .	12
6.1.3.1 addToBestScores() . . . . .	12
6.1.3.2 getBestScores() . . . . .	13
6.1.3.3 getLibName() . . . . .	13
6.1.3.4 getScore() . . . . .	13
6.1.3.5 loadFromFile() [1/2] . . . . .	13
6.1.3.6 loadFromFile() [2/2] . . . . .	14
6.1.3.7 render() . . . . .	14
6.1.3.8 saveToFile() [1/2] . . . . .	14
6.1.3.9 saveToFile() [2/2] . . . . .	15
6.1.3.10 setPlayerName() . . . . .	15
6.2 Arcade::Exceptions::ArcadeException Class Reference . . . . .	15
6.2.1 Detailed Description . . . . .	16
6.2.2 Constructor & Destructor Documentation . . . . .	16
6.2.2.1 ArcadeException() . . . . .	16
6.2.3 Member Function Documentation . . . . .	16
6.2.3.1 getComponent() . . . . .	16
6.3 Arcade::Exceptions::BadFileException Class Reference . . . . .	17

6.3.1 Detailed Description . . . . .	17
6.3.2 Constructor & Destructor Documentation . . . . .	17
6.3.2.1 BadFileException() . . . . .	17
6.4 Arcade::Exceptions::BadInstanciationException Class Reference . . . . .	18
6.4.1 Detailed Description . . . . .	18
6.4.2 Constructor & Destructor Documentation . . . . .	18
6.4.2.1 BadInstanciationException() . . . . .	18
6.5 Arcade::Games::Centipede Class Reference . . . . .	18
6.5.1 Member Function Documentation . . . . .	19
6.5.1.1 update() . . . . .	19
6.6 Arcade::Core Class Reference . . . . .	19
6.6.1 Detailed Description . . . . .	20
6.6.2 Constructor & Destructor Documentation . . . . .	20
6.6.2.1 Core() . . . . .	20
6.7 Arcade::Display::IDisplayModule Class Reference . . . . .	20
6.7.1 Detailed Description . . . . .	22
6.7.2 Member Enumeration Documentation . . . . .	22
6.7.2.1 Colors . . . . .	22
6.7.2.2 Keys . . . . .	22
6.7.3 Member Function Documentation . . . . .	23
6.7.3.1 getDelta() . . . . .	23
6.7.3.2 getKeyCode() . . . . .	23
6.7.3.3 getLibName() . . . . .	24
6.7.3.4 isKeyPressed() . . . . .	24
6.7.3.5 isKeyPressedOnce() . . . . .	24
6.7.3.6 isOpen() . . . . .	25
6.7.3.7 putCircle() . . . . .	25
6.7.3.8 putFillCircle() . . . . .	25
6.7.3.9 putFillRect() . . . . .	26
6.7.3.10 putLine() . . . . .	26
6.7.3.11 putPixel() . . . . .	26
6.7.3.12 putRect() . . . . .	27
6.7.3.13 putText() . . . . .	27
6.7.3.14 setColor() . . . . .	28
6.7.3.15 shouldBeRestarted() . . . . .	28
6.7.3.16 shouldExit() . . . . .	28
6.7.3.17 shouldGoToMenu() . . . . .	29
6.7.3.18 switchToNextGame() . . . . .	29
6.7.3.19 switchToNextLib() . . . . .	29
6.7.3.20 switchToPreviousGame() . . . . .	30
6.7.3.21 switchToPreviousLib() . . . . .	30
6.8 Arcade::Games::IGameModule Class Reference . . . . .	30

6.8.1 Detailed Description	31
6.8.2 Member Function Documentation	31
6.8.2.1 getBestScores()	31
6.8.2.2 getLibName()	32
6.8.2.3 getScore()	32
6.8.2.4 loadFromFile() [1/2]	32
6.8.2.5 loadFromFile() [2/2]	33
6.8.2.6 render()	33
6.8.2.7 saveToFile() [1/2]	33
6.8.2.8 saveToFile() [2/2]	34
6.8.2.9 setPlayerName()	34
6.8.2.10 update()	34
6.9 Arcade::Exceptions::InvalidLibraryException Class Reference	34
6.9.1 Detailed Description	35
6.9.2 Constructor & Destructor Documentation	35
6.9.2.1 InvalidLibraryException()	35
6.10 Arcade::Display::Libcaca Class Reference	35
6.10.1 Member Function Documentation	37
6.10.1.1 getDelta()	37
6.10.1.2 getKeyCode()	37
6.10.1.3 getLibName()	38
6.10.1.4 isKeyPressed()	38
6.10.1.5 isKeyPressedOnce()	38
6.10.1.6 isOpen()	39
6.10.1.7 putCircle()	39
6.10.1.8 putFillCircle()	39
6.10.1.9 putFillRect()	40
6.10.1.10 putLine()	40
6.10.1.11 putPixel()	40
6.10.1.12 putRect()	41
6.10.1.13 putText()	41
6.10.1.14 setColor()	42
6.10.1.15 shouldBeRestarted()	42
6.10.1.16 shouldExit()	42
6.10.1.17 shouldGoToMenu()	43
6.10.1.18 switchToNextGame()	43
6.10.1.19 switchToNextLib()	43
6.10.1.20 switchToPreviousGame()	44
6.10.1.21 switchToPreviousLib()	44
6.11 Arcade::Games::Nibbler Class Reference	44
6.11.1 Member Function Documentation	45
6.11.1.1 render()	45

6.11.1.2 update()	45
6.12 Arcade::Games::Pacman Class Reference	45
6.12.1 Member Function Documentation	46
6.12.1.1 render()	46
6.12.1.2 update()	46
6.13 Arcade::Games::Qix Class Reference	46
6.13.1 Member Function Documentation	47
6.13.1.1 update()	47
6.14 Arcade::Display::SDL Class Reference	47
6.14.1 Member Function Documentation	48
6.14.1.1 getDelta()	49
6.14.1.2 getKeyCode()	49
6.14.1.3 getLibName()	49
6.14.1.4 isKeyPressed()	49
6.14.1.5 isKeyPressedOnce()	50
6.14.1.6 isOpen()	50
6.14.1.7 putCircle()	50
6.14.1.8 putFillCircle()	51
6.14.1.9 putFillRect()	51
6.14.1.10 putLine()	51
6.14.1.11 putPixel()	52
6.14.1.12 putRect()	52
6.14.1.13 putText()	53
6.14.1.14 setColor()	53
6.14.1.15 shouldBeRestarted()	53
6.14.1.16 shouldExit()	54
6.14.1.17 shouldGoToMenu()	54
6.14.1.18 switchToNextGame()	54
6.14.1.19 switchToNextLib()	55
6.14.1.20 switchToPreviousGame()	55
6.14.1.21 switchToPreviousLib()	55
6.15 Arcade::Display::SFML Class Reference	55
6.15.1 Member Function Documentation	57
6.15.1.1 getDelta()	57
6.15.1.2 getKeyCode()	57
6.15.1.3 getLibName()	57
6.15.1.4 isKeyPressed()	58
6.15.1.5 isKeyPressedOnce()	58
6.15.1.6 isOpen()	58
6.15.1.7 putCircle()	58
6.15.1.8 putFillCircle()	59
6.15.1.9 putFillRect()	59

---

6.15.1.10 putLine()	60
6.15.1.11 putPixel()	60
6.15.1.12 putRect()	60
6.15.1.13 putText()	61
6.15.1.14 setColor()	61
6.15.1.15 shouldBeRestarted()	61
6.15.1.16 shouldExit()	62
6.15.1.17 shouldGoToMenu()	62
6.15.1.18 switchToNextGame()	62
6.15.1.19 switchToNextLib()	63
6.15.1.20 switchToPreviousGame()	63
6.15.1.21 switchToPreviousLib()	63
6.16 Arcade::Games::Solarfox Class Reference	63
6.16.1 Member Function Documentation	64
6.16.1.1 update()	64
<b>Index</b>	<b>65</b>





# Chapter 1

## How to implement your own game / library

### Game

- First, you wanna create a c++ class implementing the `IGameModule` interface.
- Next you wanna compile it into a dynamic library (.so file).
- Put that library file in the `games/` folder located at the root of the arcade repository.
  - Note that it must follow the following naming expression : `lib_arcade_.*.so`, so that means a name like **`lib_arcade_$gamename.so`**.

Every class contained by the libraries located in the `games/` folder are going to be instanciated thanks to the symbol `createLib` that your library **must** contain.

```
extern "C" std::unique_ptr<Arcade::Games::IGameModule> createLib(void)
{
    return std::make_unique<MyGameModule>();
}
```

### Library

- First, you wanna create a c++ class implementing the `IDisplayModule` interface.
- Next you wanna compile it into a dynamic library (.so file).
- Put that library file in the `lib/` folder located at the root of the arcade repository.
  - Note that it must follow the following naming expression : `lib_arcade_.*.so`, so that means a name like **`lib_arcade_$libraryname.so`**.

Every class contained by the libraries located in the `lib/` folder are going to be instanciated thanks to the symbol `createLib` that your library **must** contain.

```
extern "C" std::unique_ptr<Arcade::Display::IDisplayModule> createLib(void)
{
    return std::make_unique<MyDisplayModule>();
}
```



## Chapter 2

# Namespace Index

### 2.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

<a href="#">Arcade</a>	9
<a href="#">Arcade::Display</a>	9
<a href="#">Arcade::Exceptions</a>	10
<a href="#">Arcade::Games</a>	10



## Chapter 3

# Hierarchical Index

### 3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Arcade::Core . . . . .	19
Arcade::Display::IDisplayModule . . . . .	20
Arcade::Display::Libcaca . . . . .	35
Arcade::Display::SDL . . . . .	47
Arcade::Display::SFML . . . . .	55
Arcade::Games::IGameModule . . . . .	30
Arcade::Games::AGameModule . . . . .	11
Arcade::Games::Centipede . . . . .	18
Arcade::Games::Nibbler . . . . .	44
Arcade::Games::Pacman . . . . .	45
Arcade::Games::Qix . . . . .	46
Arcade::Games::Solarfox . . . . .	63
runtime_error	
Arcade::Exceptions::ArcadeException . . . . .	15
Arcade::Exceptions::BadFileException . . . . .	17
Arcade::Exceptions::BadInstanciationException . . . . .	18
Arcade::Exceptions::InvalidLibraryException . . . . .	34



## Chapter 4

# Class Index

### 4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">Arcade::Games::AGameModule</a>	
Abstract class implementing key methods of the <a href="#">IGameModule</a> interface . . . . .	11
<a href="#">Arcade::Exceptions::ArcadeException</a>	
Base exception class for this projects' exceptions . . . . .	15
<a href="#">Arcade::Exceptions::BadFileException</a>	
Thrown when looking up to an external file that is inexistant . . . . .	17
<a href="#">Arcade::Exceptions::BadInstanciationException</a>	
Thrown when library objects failed to be instanciated . . . . .	18
<a href="#">Arcade::Games::Centipede</a>	18
<a href="#">Arcade::Core</a>	
Core class that handles all the interactions between the library modules and the game modules	19
<a href="#">Arcade::Display::IDisplayModule</a>	
Interface for the display modules used to display things . . . . .	20
<a href="#">Arcade::Games::IGameModule</a>	
Interface for the game modules used to handle games . . . . .	30
<a href="#">Arcade::Exceptions::InvalidLibraryException</a>	
Thrown when trying to use an invalid library file . . . . .	34
<a href="#">Arcade::Display::Libcaca</a>	35
<a href="#">Arcade::Games::Nibbler</a>	44
<a href="#">Arcade::Games::Pacman</a>	45
<a href="#">Arcade::Games::Qix</a>	46
<a href="#">Arcade::Display::SDL</a>	47
<a href="#">Arcade::Display::SFML</a>	55
<a href="#">Arcade::Games::Solarfox</a>	63





## Chapter 5

# Namespace Documentation

### 5.1 Arcade Namespace Reference

#### Namespaces

- [Display](#)
- [Exceptions](#)
- [Games](#)

#### Classes

- class [Core](#)  
*[Core](#) class that handles all the interactions between the library modules and the game modules.*

#### 5.1.1 Detailed Description

Default namespace for the project.

### 5.2 Arcade::Display Namespace Reference

#### Classes

- class [IDisplayModule](#)  
*Interface for the display modules used to display things.*
- class [Libcaca](#)
- class [SDL](#)
- class [SFML](#)

#### 5.2.1 Detailed Description

Contains elements related to the display libraries of the [Arcade](#) project.

## 5.3 Arcade::Exceptions Namespace Reference

### Classes

- class [ArcadeException](#)  
*Base exception class for this projects' exceptions.*
- class [BadFileException](#)  
*Thrown when looking up to an external file that is inexistant.*
- class [BadInstanciationException](#)  
*Thrown when library objects failed to be instanciated.*
- class [InvalidLibraryException](#)  
*Thrown when trying to use an invalid library file.*

#### 5.3.1 Detailed Description

Contains a loadout of exceptions that are used in the [Arcade](#) project.

## 5.4 Arcade::Games Namespace Reference

### Classes

- class [AGameModule](#)  
*Abstract class implementing key methods of the [IGameModule](#) interface.*
- class [Centipede](#)
- class [IGameModule](#)  
*Interface for the game modules used to handle games.*
- class [Nibbler](#)
- class [Pacman](#)
- class [Qix](#)
- class [Solarfox](#)

#### 5.4.1 Detailed Description

Contains elements related to the game libraries of the [Arcade](#) project.

## Chapter 6

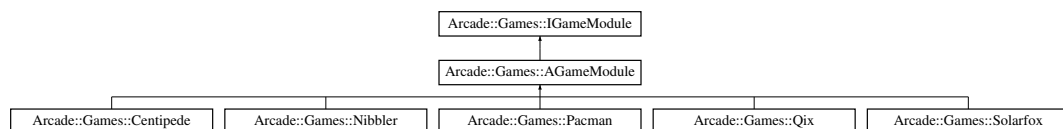
# Class Documentation

### 6.1 Arcade::Games::AGameModule Class Reference

Abstract class implementing key methods of the [IGameModule](#) interface.

```
#include <AGameModule.hpp>
```

Inheritance diagram for Arcade::Games::AGameModule:



#### Public Member Functions

- [AGameModule](#) (std::string const &libname)  
*Construct a new [AGameModule](#) object.*
- bool [loadFromFile](#) (const std::string &filepath) final  
*Loads highscores from a file.*
- bool [loadFromFile](#) () final  
*Loads highscores from the default save file.*
- bool [saveToFile](#) (const std::string &filepath) const final  
*Saves highscores to a file.*
- bool [saveToFile](#) () const final  
*Saves highscores from the default save file.*
- void [setPlayerName](#) (const std::string &name) final  
*Sets the player name.*
- std::pair< std::string, int > [getScore](#) () const final  
*Gets the current score.*
- std::vector< std::pair< std::string, int > > [getBestScores](#) () const final  
*Gets the best 16 scores.*
- void [render](#) ([Arcade::Display::IDisplayModule](#) &lib) const override
- const std::string & [getLibName](#) () const final  
*Gets the library name.*

## Protected Member Functions

- void [addToBestScores](#) (int nb)  
*Adds a score to the scoreboard.*

## Protected Attributes

- int [\\_currentScore](#)  
*The current score of the active game session.*

### 6.1.1 Detailed Description

Abstract class implementing key methods of the [IGameModule](#) interface.

### 6.1.2 Constructor & Destructor Documentation

#### 6.1.2.1 AGameModule()

```
Arcade::Games::AGameModule::AGameModule (
    std::string const & libname )
```

Construct a new [AGameModule](#) object.

#### Parameters

<i>libname</i>	The library's name
----------------	--------------------

### 6.1.3 Member Function Documentation

#### 6.1.3.1 addToBestScores()

```
void Arcade::Games::AGameModule::addToBestScores (
    int nb ) [protected]
```

Adds a score to the scoreboard.

#### Parameters

<i>nb</i>	The score value
-----------	-----------------

#### 6.1.3.2 getBestScores()

```
std::vector<std::pair<std::string, int> > Arcade::Games::AGameModule::getBestScores ( ) const  
[final], [virtual]
```

Gets the best 16 scores.

##### Returns

std::vector<std::pair<std::string, int>> Vector of [name, score] value pairs

Implements [Arcade::Games::IGameModule](#).

#### 6.1.3.3 getLibName()

```
const std::string& Arcade::Games::AGameModule::getLibName ( ) const [final], [virtual]
```

Gets the library name.

##### Returns

The library's name

Implements [Arcade::Games::IGameModule](#).

#### 6.1.3.4 getScore()

```
std::pair<std::string, int> Arcade::Games::AGameModule::getScore ( ) const [final], [virtual]
```

Gets the current score.

##### Returns

std::pair<std::string, int> [Name, score] value pairs

Implements [Arcade::Games::IGameModule](#).

#### 6.1.3.5 loadFromFile() [1/2]

```
bool Arcade::Games::AGameModule::loadFromFile (  
    const std::string & filepath ) [final], [virtual]
```

Loads highscores from a file.

**Parameters**

<i>filepath</i>	The file path
-----------------	---------------

**Returns**

true Highscores were loaded  
false An error occurred

Implements [Arcade::Games::IGameModule](#).

**6.1.3.6 loadFromFile()** [2/2]

```
bool Arcade::Games::AGameModule::loadFromFile ( ) [final], [virtual]
```

Loads highscores from the default save file.

**Returns**

true Highscores were loaded  
false An error occurred

Implements [Arcade::Games::IGameModule](#).

**6.1.3.7 render()**

```
void Arcade::Games::AGameModule::render (
    Arcade::Display::IDisplayModule & lib ) const [override], [virtual]
```

Default game implementation (out of order)

Implements [Arcade::Games::IGameModule](#).

Reimplemented in [Arcade::Games::Nibbler](#), and [Arcade::Games::Pacman](#).

**6.1.3.8 saveToFile()** [1/2]

```
bool Arcade::Games::AGameModule::saveToFile (
    const std::string & filepath ) const [final], [virtual]
```

Saves highscores to a file.

## Parameters

<i>filepath</i>	The file path
-----------------	---------------

## Returns

true Highscores were saved  
false An error occurred

Implements [Arcade::Games::IGameModule](#).

## 6.1.3.9 saveToFile() [2/2]

```
bool Arcade::Games::AGameModule::saveToFile ( ) const [final], [virtual]
```

Saves highscores from the default save file.

## Returns

true Highscores were saved  
false An error occurred

Implements [Arcade::Games::IGameModule](#).

## 6.1.3.10 setPlayerName()

```
void Arcade::Games::AGameModule::setPlayerName (
    const std::string & name ) [final], [virtual]
```

Sets the player name.

## Parameters

<i>name</i>	The player name
-------------	-----------------

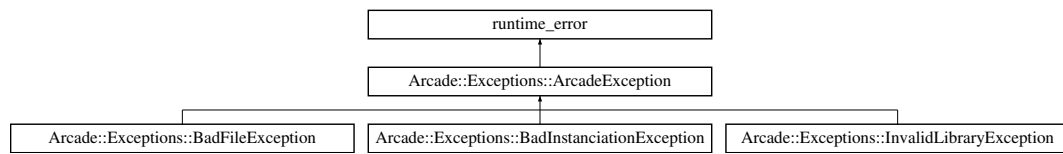
Implements [Arcade::Games::IGameModule](#).

## 6.2 Arcade::Exceptions::ArcadeException Class Reference

Base exception class for this projects' exceptions.

```
#include <ArcadeException.hpp>
```

Inheritance diagram for `Arcade::Exceptions::ArcadeException`:



## Public Member Functions

- [ArcadeException](#) (`std::string const &message`, `std::string const &component`)  
Construct a new [Arcade](#) Exception object.
- `std::string const &getComponent` (`void`) `const noexcept`  
Gets the name of component where the exception occurred.

### 6.2.1 Detailed Description

Base exception class for this projects' exceptions.

### 6.2.2 Constructor & Destructor Documentation

#### 6.2.2.1 `ArcadeException()`

```

Arcade::Exceptions::ArcadeException::ArcadeException (
    std::string const & message,
    std::string const & component )
  
```

Construct a new [Arcade](#) Exception object.

#### Parameters

<i>message</i>	Message explaining the problem.
<i>component</i>	Additional information on where the problem occurred.

### 6.2.3 Member Function Documentation

#### 6.2.3.1 `getComponent()`

```

std::string const& Arcade::Exceptions::ArcadeException::getComponent (
    void ) const [noexcept]
  
```

Gets the name of component where the exception occurred.



**Returns**

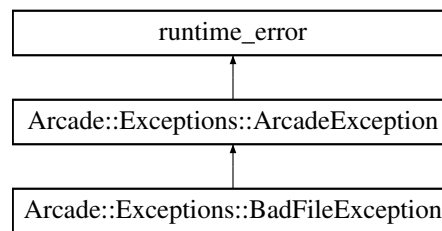
The component name

## 6.3 Arcade::Exceptions::BadFileException Class Reference

Thrown when looking up to an external file that is inexistant.

```
#include <BadFileException.hpp>
```

Inheritance diagram for Arcade::Exceptions::BadFileException:

**Public Member Functions**

- [BadFileException](#) (std::string const &message, std::string const &component)  
*Construct a new Bad File Exception object.*

### 6.3.1 Detailed Description

Thrown when looking up to an external file that is inexistant.

### 6.3.2 Constructor & Destructor Documentation

#### 6.3.2.1 BadFileException()

```
Arcade::Exceptions::BadFileException::BadFileException (
    std::string const & message,
    std::string const & component )
```

Construct a new Bad File Exception object.

**Parameters**

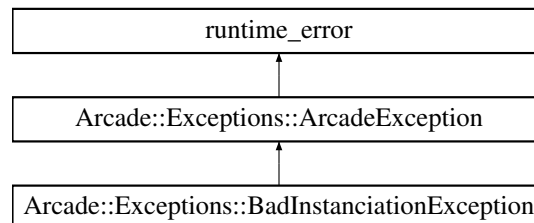
<i>message</i>	Message explaining the problem.
<i>component</i>	Additional information on where the problem occurred.

## 6.4 Arcade::Exceptions::BadInstanciationException Class Reference

Thrown when library objects failed to be instantiated.

```
#include <BadInstanciationException.hpp>
```

Inheritance diagram for Arcade::Exceptions::BadInstanciationException:



### Public Member Functions

- [BadInstanciationException](#) (std::string const &message, std::string const &component)  
*Construct a new Bad InstanciationException object.*

#### 6.4.1 Detailed Description

Thrown when library objects failed to be instantiated.

#### 6.4.2 Constructor & Destructor Documentation

##### 6.4.2.1 BadInstanciationException()

```
Arcade::Exceptions::BadInstanciationException::BadInstanciationException (
    std::string const & message,
    std::string const & component )
```

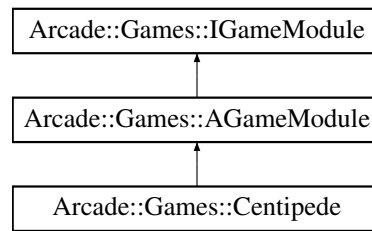
Construct a new Bad InstanciationException object.

#### Parameters

<i>message</i>	Message explaining the problem.
<i>component</i>	Additional information on where the problem occurred.

## 6.5 Arcade::Games::Centipede Class Reference

Inheritance diagram for Arcade::Games::Centipede:



## Public Member Functions

- void [reset](#) () final  
*Resets and restarts the game.*
- void [update](#) (const [Arcade::Display::IDisplayModule](#) &lib) final  
*Updates the game.*

## Additional Inherited Members

### 6.5.1 Member Function Documentation

#### 6.5.1.1 update()

```
void Arcade::Games::Centipede::update (
    const Arcade::Display::IDisplayModule & lib ) [final], [virtual]
```

Updates the game.

#### Parameters

<i>lib</i>	The display module that will be used to get events that occurred
------------	--

Implements [Arcade::Games::IGameModule](#).

## 6.6 Arcade::Core Class Reference

[Core](#) class that handles all the interactions between the library modules and the game modules.

```
#include <Core.hpp>
```

## Public Member Functions

- [Core](#) (const std::string &startLibraryPath)  
*Construct a new [Core](#) object.*
- void [play](#) ()  
*Starts the arcade program.*

### 6.6.1 Detailed Description

[Core](#) class that handles all the interactions between the library modules and the game modules.

### 6.6.2 Constructor & Destructor Documentation

#### 6.6.2.1 Core()

```
Arcade::Core::Core (
    const std::string & startLibraryPath ) [explicit]
```

Construct a new [Core](#) object.

##### Parameters

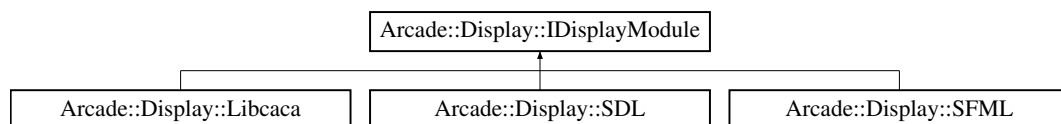
<i>startLibraryPath</i>	The library path that will be first used when the program will be started.
-------------------------	--

## 6.7 Arcade::Display::IDisplayModule Class Reference

Interface for the display modules used to display things.

```
#include <IDisplayModule.hpp>
```

Inheritance diagram for Arcade::Display::IDisplayModule:



### Public Types

- enum [Colors](#) {  
[DEFAULT](#), [BLACK](#), [RED](#), [GREEN](#),  
[YELLOW](#), [BLUE](#), [MAGENTA](#), [CYAN](#),  
[LIGHT\\_GRAY](#), [DARK\\_GRAY](#), [LIGHT\\_RED](#), [LIGHT\\_GREEN](#),  
[LIGHT\\_YELLOW](#), [LIGHT\\_BLUE](#), [LIGHT\\_MAGENTA](#), [LIGHT\\_CYAN](#),  
[WHITE](#), [COLORS\\_END](#) }

*Available colors.*

- enum [Keys](#) {  
[LEFT](#), [RIGHT](#), [UP](#), [DOWN](#),  
[Z](#), [Q](#), [S](#), [D](#),  
[A](#), [E](#), [W](#), [X](#),  
[SPACE](#), [ESCAPE](#), [J](#), [K](#),  
[U](#), [I](#), [M](#), [R](#),  
[ENTER](#), [KEYS\\_END](#) }

*Available keys.*

## Public Member Functions

- virtual void [reset](#) ()=0  
*Resets the library.*
- virtual void [open](#) ()=0  
*Opens / initializes the window.*
- virtual bool [isOpen](#) () const =0  
*Check window status.*
- virtual bool [switchToNextLib](#) () const =0  
*Checks whether you need to change the current display library.*
- virtual bool [switchToPreviousLib](#) () const =0  
*Checks whether you need to change the current display library.*
- virtual bool [switchToNextGame](#) () const =0  
*Checks whether you need to change the current game library.*
- virtual bool [switchToPreviousGame](#) () const =0  
*Checks whether you need to change the current game library.*
- virtual bool [shouldBeRestarted](#) () const =0  
*Checks whether you need to restart the current game.*
- virtual bool [shouldGoToMenu](#) () const =0  
*Checks whether you need to go back to the menu.*
- virtual bool [shouldExit](#) () const =0  
*Checks whether you need to exit the program.*
- virtual bool [isKeyPressed](#) (IDisplayModule::Keys) const =0  
*Checks whether the current key is being pressed.*
- virtual bool [isKeyPressedOnce](#) (IDisplayModule::Keys) const =0  
*Checks whether the current key was pressed during the last frame.*
- virtual float [getDelta](#) () const =0  
*Gets the number of frames since last update.*
- virtual void [clear](#) () const =0  
*Clears the canvas.*
- virtual void [update](#) ()=0  
*Runs an update over the events that occurred.*
- virtual void [render](#) () const =0  
*Renders the canvas.*
- virtual char [getKeyCode](#) () const =0  
*Gets the last pressed character from the keyboard.*
- virtual void [setColor](#) (IDisplayModule::Colors col)=0  
*Defines the color of the elements that will be drawn.*
- virtual void [putPixel](#) (float x, float y) const =0  
*Displays a pixel.*
- virtual void [putLine](#) (float x1, float y1, float x2, float y2) const =0  
*Displays a line.*
- virtual void [putRect](#) (float x, float y, float w, float h) const =0  
*Displays a rectangle.*
- virtual void [putFillRect](#) (float x, float y, float w, float h) const =0  
*Displays a filled rectangle.*
- virtual void [putCircle](#) (float x, float y, float rad) const =0  
*Displays a circle.*
- virtual void [putFillCircle](#) (float x, float y, float rad) const =0  
*Displays a filled circle.*
- virtual void [putText](#) (const std::string &text, unsigned int size, float x, float y) const =0  
*Displays text.*
- virtual const std::string & [getLibName](#) () const =0  
*Gets the library name.*

### 6.7.1 Detailed Description

Interface for the display modules used to display things.

### 6.7.2 Member Enumeration Documentation

#### 6.7.2.1 Colors

enum `Arcade::Display::IDisplayModule::Colors`

Available colors.

##### Enumerator

DEFAULT	The color the window clears to.
BLACK	Black color.
RED	Red color.
GREEN	Green color.
YELLOW	Yellow color.
BLUE	Blue color.
MAGENTA	Magenta color.
CYAN	Cyan color.
LIGHT_GRAY	Light gray color.
DARK_GRAY	Dark gray color.
LIGHT_RED	Light red color.
LIGHT_GREEN	Light green color.
LIGHT_YELLOW	Light yellow color.
LIGHT_BLUE	Light blue color.
LIGHT_MAGENTA	Light magenta color.
LIGHT_CYAN	Light cyan color.
WHITE	White color.
COLORS_END	Color count.

#### 6.7.2.2 Keys

enum `Arcade::Display::IDisplayModule::Keys`

Available keys.

##### Enumerator

LEFT	Left key.
RIGHT	Right key.

## Enumerator

UP	Up key.
DOWN	Down key.
Z	Z key.
Q	Q key.
S	S key.
D	D key.
A	A key.
E	E key.
W	W key.
X	X key.
SPACE	Space key.
ESCAPE	Backspace key.
J	J key.
K	K key.
U	U key.
I	I key.
M	M key.
R	R key.
ENTER	Return key.
KEYS_END	Key count.

## 6.7.3 Member Function Documentation

## 6.7.3.1 getDelta()

```
virtual float Arcade::Display::IDisplayModule::getDelta ( ) const [pure virtual]
```

Gets the number of frames since last update.

## Returns

float Frame count

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

## 6.7.3.2 getKeyCode()

```
virtual char Arcade::Display::IDisplayModule::getKeyCode ( ) const [pure virtual]
```

Gets the last pressed character from the keyboard.

## Returns

\0 if nothing was pressed, \b if backspace was pressed, \n if return was pressed, otherwise, a character.

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

### 6.7.3.3 getLibName()

```
virtual const std::string& Arcade::Display::IDisplayModule::getLibName ( ) const [pure virtual]
```

Gets the library name.

#### Returns

The library's name

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

### 6.7.3.4 isKeyPressed()

```
virtual bool Arcade::Display::IDisplayModule::isKeyPressed (
    IDisplayModule::Keys ) const [pure virtual]
```

Checks whether the current key is being pressed.

#### Returns

true Key is pressed  
false Key is not pressed

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

### 6.7.3.5 isKeyPressedOnce()

```
virtual bool Arcade::Display::IDisplayModule::isKeyPressedOnce (
    IDisplayModule::Keys ) const [pure virtual]
```

Checks whether the current key was pressed during the last frame.

#### Returns

true Key is pressed  
false Key is not pressed

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).



#### 6.7.3.6 isOpen()

```
virtual bool Arcade::Display::IDisplayModule::isOpen ( ) const [pure virtual]
```

Check window status.

##### Returns

true Window is open  
false Window is closed

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

#### 6.7.3.7 putCircle()

```
virtual void Arcade::Display::IDisplayModule::putCircle (
    float x,
    float y,
    float rad ) const [pure virtual]
```

Displays a circle.

##### Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>rad</i>	Radius of the circle

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

#### 6.7.3.8 putFillColor()

```
virtual void Arcade::Display::IDisplayModule::putFillColor (
    float x,
    float y,
    float rad ) const [pure virtual]
```

Displays a filled circle.

##### Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>rad</i>	Radius of the circle

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

### 6.7.3.9 putFillRect()

```
virtual void Arcade::Display::IDisplayModule::putFillRect (
    float x,
    float y,
    float w,
    float h ) const [pure virtual]
```

Displays a filled rectangle.

#### Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>w</i>	Width of the rectangle
<i>h</i>	Height of the rectangle

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

### 6.7.3.10 putLine()

```
virtual void Arcade::Display::IDisplayModule::putLine (
    float x1,
    float y1,
    float x2,
    float y2 ) const [pure virtual]
```

Displays a line.

#### Parameters

<i>x1</i>	X coordinates for the first point
<i>y1</i>	Y coordinates for the first point
<i>x2</i>	X coordinates for the second point
<i>y2</i>	Y coordinates for the second point

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

### 6.7.3.11 putPixel()

```
virtual void Arcade::Display::IDisplayModule::putPixel (
    float x,
    float y ) const [pure virtual]
```

Displays a pixel.

## Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

## 6.7.3.12 putRect()

```
virtual void Arcade::Display::IDisplayModule::putRect (
    float x,
    float y,
    float w,
    float h ) const [pure virtual]
```

Displays a rectangle.

## Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>w</i>	Width of the rectangle
<i>h</i>	Height of the rectangle

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

## 6.7.3.13 putText()

```
virtual void Arcade::Display::IDisplayModule::putText (
    const std::string & text,
    unsigned int size,
    float x,
    float y ) const [pure virtual]
```

Displays text.

## Parameters

<i>text</i>	The text content
<i>size</i>	The text size
<i>x</i>	X coordinates
<i>y</i>	Y coordinates

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

#### 6.7.3.14 setColor()

```
virtual void Arcade::Display::IDisplayModule::setColor (
    IDisplayModule::Colors col ) [pure virtual]
```

Defines the color of the elements that will be drawn.

##### Parameters

<i>col</i>	The color
------------	-----------

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

#### 6.7.3.15 shouldBeRestarted()

```
virtual bool Arcade::Display::IDisplayModule::shouldBeRestarted ( ) const [pure virtual]
```

Checks whether you need to restart the current game.

##### Returns

true Restart the game  
false Do nothing

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

#### 6.7.3.16 shouldExit()

```
virtual bool Arcade::Display::IDisplayModule::shouldExit ( ) const [pure virtual]
```

Checks whether you need to exit the program.

##### Returns

true Exit the program  
false Do nothing

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

#### 6.7.3.17 shouldGoToMenu()

```
virtual bool Arcade::Display::IDisplayModule::shouldGoToMenu ( ) const [pure virtual]
```

Checks whether you need to go back to the menu.

##### Returns

true Go back to menu  
false Do nothing

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

#### 6.7.3.18 switchToNextGame()

```
virtual bool Arcade::Display::IDisplayModule::switchToNextGame ( ) const [pure virtual]
```

Checks whether you need to change the current game library.

##### Returns

true Switch to next available library  
false Do nothing

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

#### 6.7.3.19 switchToNextLib()

```
virtual bool Arcade::Display::IDisplayModule::switchToNextLib ( ) const [pure virtual]
```

Checks whether you need to change the current display library.

##### Returns

true Switch to next available library  
false Do nothing

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

### 6.7.3.20 switchToPreviousGame()

```
virtual bool Arcade::Display::IDisplayModule::switchToPreviousGame ( ) const [pure virtual]
```

Checks whether you need to change the current game library.

#### Returns

true Switch to previous available library  
false Do nothing

Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

### 6.7.3.21 switchToPreviousLib()

```
virtual bool Arcade::Display::IDisplayModule::switchToPreviousLib ( ) const [pure virtual]
```

Checks whether you need to change the current display library.

#### Returns

true Switch to previous available library  
false Do nothing

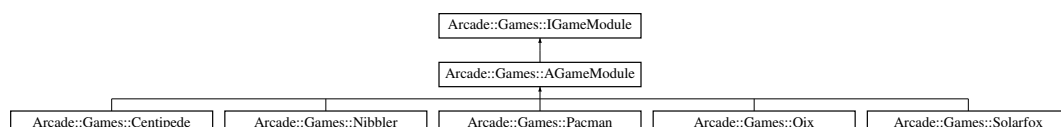
Implemented in [Arcade::Display::SDL](#), [Arcade::Display::Libcaca](#), and [Arcade::Display::SFML](#).

## 6.8 Arcade::Games::IGameModule Class Reference

Interface for the game modules used to handle games.

```
#include <IGameModule.hpp>
```

Inheritance diagram for Arcade::Games::IGameModule:



## Public Member Functions

- virtual void [reset](#) ()=0  
*Resets and restarts the game.*
- virtual bool [loadFromFile](#) (const std::string &filepath)=0  
*Loads highscores from a file.*
- virtual bool [loadFromFile](#) ()=0  
*Loads highscores from the default save file.*
- virtual bool [saveToFile](#) (const std::string &filepath) const =0  
*Saves highscores to a file.*
- virtual bool [saveToFile](#) () const =0  
*Saves highscores from the default save file.*
- virtual void [setPlayerName](#) (const std::string &name)=0  
*Sets the player name.*
- virtual std::pair< std::string, int > [getScore](#) () const =0  
*Gets the current score.*
- virtual std::vector< std::pair< std::string, int > > [getBestScores](#) () const =0  
*Gets the best 16 scores.*
- virtual void [update](#) (const [Arcade::Display::IDisplayModule](#) &lib)=0  
*Updates the game.*
- virtual void [render](#) ([Arcade::Display::IDisplayModule](#) &lib) const =0  
*Renders the game on the display module.*
- virtual const std::string & [getLibName](#) () const =0  
*Gets the library name.*

### 6.8.1 Detailed Description

Interface for the game modules used to handle games.

### 6.8.2 Member Function Documentation

#### 6.8.2.1 [getBestScores\(\)](#)

```
virtual std::vector<std::pair<std::string, int> > Arcade::Games::IGameModule::getBestScores (
) const [pure virtual]
```

Gets the best 16 scores.

#### Returns

std::vector<std::pair<std::string, int>> Vector of [name, score] value pairs

Implemented in [Arcade::Games::AGameModule](#).

### 6.8.2.2 getLibName()

```
virtual const std::string& Arcade::Games::IGameModule::getLibName ( ) const [pure virtual]
```

Gets the library name.

#### Returns

The library's name

Implemented in [Arcade::Games::AGameModule](#).

### 6.8.2.3 getScore()

```
virtual std::pair<std::string, int> Arcade::Games::IGameModule::getScore ( ) const [pure virtual]
```

Gets the current score.

#### Returns

std::pair<std::string, int> [Name, score] value pairs

Implemented in [Arcade::Games::AGameModule](#).

### 6.8.2.4 loadFromFile() [1/2]

```
virtual bool Arcade::Games::IGameModule::loadFromFile (
    const std::string & filepath ) [pure virtual]
```

Loads highscores from a file.

#### Parameters

<i>filepath</i>	The file path
-----------------	---------------

#### Returns

true Highscores were loaded  
false An error occurred

Implemented in [Arcade::Games::AGameModule](#).



### 6.8.2.5 loadFromFile() [2/2]

```
virtual bool Arcade::Games::IGameModule::loadFromFile ( ) [pure virtual]
```

Loads highscores from the default save file.

#### Returns

true Highscores were loaded  
false An error occurred

Implemented in [Arcade::Games::AGameModule](#).

### 6.8.2.6 render()

```
virtual void Arcade::Games::IGameModule::render (
    Arcade::Display::IDisplayModule & lib ) const [pure virtual]
```

Renders the game on the display module.

#### Parameters

<i>lib</i>	The display module that will be used to put things on a canvas.
------------	---

Implemented in [Arcade::Games::AGameModule](#), [Arcade::Games::Nibbler](#), and [Arcade::Games::Pacman](#).

### 6.8.2.7 saveToFile() [1/2]

```
virtual bool Arcade::Games::IGameModule::saveToFile (
    const std::string & filepath ) const [pure virtual]
```

Saves highscores to a file.

#### Parameters

<i>filepath</i>	The file path
-----------------	---------------

#### Returns

true Highscores were saved  
false An error occurred

Implemented in [Arcade::Games::AGameModule](#).

**6.8.2.8 saveToFile()** [2/2]

```
virtual bool Arcade::Games::IGameModule::saveToFile ( ) const [pure virtual]
```

Saves highscores from the default save file.

**Returns**

true Highscores were saved  
false An error occurred

Implemented in [Arcade::Games::AGameModule](#).

**6.8.2.9 setPlayerName()**

```
virtual void Arcade::Games::IGameModule::setPlayerName (
    const std::string & name ) [pure virtual]
```

Sets the player name.

**Parameters**

<i>name</i>	The player name
-------------	-----------------

Implemented in [Arcade::Games::AGameModule](#).

**6.8.2.10 update()**

```
virtual void Arcade::Games::IGameModule::update (
    const Arcade::Display::IDisplayModule & lib ) [pure virtual]
```

Updates the game.

**Parameters**

<i>lib</i>	The display module that will be used to get events that occurred
------------	--

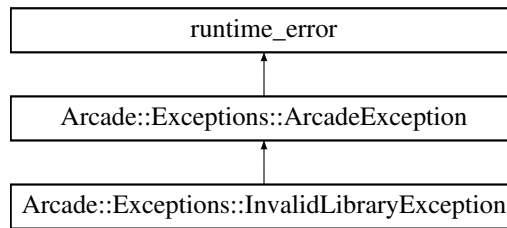
Implemented in [Arcade::Games::Nibbler](#), [Arcade::Games::Pacman](#), [Arcade::Games::Centipede](#), [Arcade::Games::Qix](#), and [Arcade::Games::Solarfox](#).

**6.9 Arcade::Exceptions::InvalidLibraryException Class Reference**

Thrown when trying to use an invalid library file.

```
#include <InvalidLibraryException.hpp>
```

Inheritance diagram for Arcade::Exceptions::InvalidLibraryException:



## Public Member Functions

- [InvalidLibraryException](#) (std::string const &message, std::string const &component)  
Construct a new Invalid Library Exception object.

### 6.9.1 Detailed Description

Thrown when trying to use an invalid library file.

### 6.9.2 Constructor & Destructor Documentation

#### 6.9.2.1 InvalidLibraryException()

```

Arcade::Exceptions::InvalidLibraryException::InvalidLibraryException (
    std::string const & message,
    std::string const & component )

```

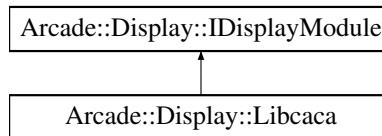
Construct a new Invalid Library Exception object.

#### Parameters

<i>message</i>	Message explaining the problem.
<i>component</i>	Additional information on where the problem occurred.

## 6.10 Arcade::Display::Libcaca Class Reference

Inheritance diagram for Arcade::Display::Libcaca:



## Public Member Functions

- void **reset** () final  
*Resets the library.*
- void **open** () final  
*Opens / initializes the window.*
- bool **isOpen** () const final  
*Check window status.*
- bool **switchToNextLib** () const final  
*Checks whether you need to change the current display library.*
- bool **switchToPreviousLib** () const final  
*Checks whether you need to change the current display library.*
- bool **switchToNextGame** () const final  
*Checks whether you need to change the current game library.*
- bool **switchToPreviousGame** () const final  
*Checks whether you need to change the current game library.*
- bool **shouldBeRestarted** () const final  
*Checks whether you need to restart the current game.*
- bool **shouldGoToMenu** () const final  
*Checks whether you need to go back to the menu.*
- bool **shouldExit** () const final  
*Checks whether you need to exit the program.*
- bool **isKeyPressed** (IDisplayModule::Keys) const final  
*Checks whether the current key is being pressed.*
- bool **isKeyPressedOnce** (IDisplayModule::Keys) const final  
*Checks whether the current key was pressed during the last frame.*
- float **getDelta** () const final  
*Gets the number of frames since last update.*
- void **clear** () const final  
*Clears the canvas.*
- void **update** () final  
*Runs an update over the events that occurred.*
- void **render** () const final  
*Renders the canvas.*
- char **getKeyCode** () const final  
*Gets the last pressed character from the keyboard.*
- void **setColor** (IDisplayModule::Colors col) final  
*Defines the color of the elements that will be drawn.*
- void **putPixel** (float x, float y) const final  
*Displays a pixel.*
- void **putLine** (float x1, float y1, float x2, float y2) const final  
*Displays a line.*
- void **putRect** (float x, float y, float w, float h) const final  
*Displays a rectangle.*

- void [putFillRect](#) (float x, float y, float w, float h) const final  
*Displays a filled rectangle.*
- void [putCircle](#) (float x, float y, float rad) const final  
*Displays a circle.*
- void [putFillCircle](#) (float x, float y, float rad) const final  
*Displays a filled circle.*
- void [putText](#) (const std::string &text, unsigned int size, float x, float y) const final  
*Displays text.*
- const std::string & [getLibName](#) () const final  
*Gets the library name.*

## Additional Inherited Members

### 6.10.1 Member Function Documentation

#### 6.10.1.1 [getDelta\(\)](#)

```
float Arcade::Display::Libcaca::getDelta ( ) const [final], [virtual]
```

Gets the number of frames since last update.

##### Returns

float Frame count

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.2 [getKeyCode\(\)](#)

```
char Arcade::Display::Libcaca::getKeyCode ( ) const [final], [virtual]
```

Gets the last pressed character from the keyboard.

##### Returns

\0 if nothing was pressed, \b if backspace was pressed, \n if return was pressed, otherwise, a character.

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.3 getLibName()

```
const std::string& Arcade::Display::Libcaca::getLibName ( ) const [final], [virtual]
```

Gets the library name.

##### Returns

The library's name

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.4 isKeyPressed()

```
bool Arcade::Display::Libcaca::isKeyPressed (
    IDisplayModule::Keys ) const [final], [virtual]
```

Checks whether the current key is being pressed.

##### Returns

true Key is pressed  
false Key is not pressed

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.5 isKeyPressedOnce()

```
bool Arcade::Display::Libcaca::isKeyPressedOnce (
    IDisplayModule::Keys ) const [final], [virtual]
```

Checks whether the current key was pressed during the last frame.

##### Returns

true Key is pressed  
false Key is not pressed

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.6 isOpen()

```
bool Arcade::Display::Libcaca::isOpen ( ) const [final], [virtual]
```

Check window status.

##### Returns

true Window is open  
false Window is closed

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.7 putCircle()

```
void Arcade::Display::Libcaca::putCircle (
    float x,
    float y,
    float rad ) const [final], [virtual]
```

Displays a circle.

##### Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>rad</i>	Radius of the circle

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.8 putFillCircle()

```
void Arcade::Display::Libcaca::putFillCircle (
    float x,
    float y,
    float rad ) const [final], [virtual]
```

Displays a filled circle.

##### Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>rad</i>	Radius of the circle

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.9 putFillRect()

```
void Arcade::Display::Libcaca::putFillRect (
    float x,
    float y,
    float w,
    float h ) const [final], [virtual]
```

Displays a filled rectangle.

##### Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>w</i>	Width of the rectangle
<i>h</i>	Height of the rectangle

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.10 putLine()

```
void Arcade::Display::Libcaca::putLine (
    float x1,
    float y1,
    float x2,
    float y2 ) const [final], [virtual]
```

Displays a line.

##### Parameters

<i>x1</i>	X coordinates for the first point
<i>y1</i>	Y coordinates for the first point
<i>x2</i>	X coordinates for the second point
<i>y2</i>	Y coordinates for the second point

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.11 putPixel()

```
void Arcade::Display::Libcaca::putPixel (
    float x,
    float y ) const [final], [virtual]
```

Displays a pixel.



## Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates

Implements [Arcade::Display::IDisplayModule](#).

## 6.10.1.12 putRect()

```
void Arcade::Display::Libcaca::putRect (
    float x,
    float y,
    float w,
    float h ) const [final], [virtual]
```

Displays a rectangle.

## Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>w</i>	Width of the rectangle
<i>h</i>	Height of the rectangle

Implements [Arcade::Display::IDisplayModule](#).

## 6.10.1.13 putText()

```
void Arcade::Display::Libcaca::putText (
    const std::string & text,
    unsigned int size,
    float x,
    float y ) const [final], [virtual]
```

Displays text.

## Parameters

<i>text</i>	The text content
<i>size</i>	The text size
<i>x</i>	X coordinates
<i>y</i>	Y coordinates

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.14 setColor()

```
void Arcade::Display::Libcaca::setColor (
    IDisplayModule::Colors col ) [final], [virtual]
```

Defines the color of the elements that will be drawn.

##### Parameters

<i>col</i>	The color
------------	-----------

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.15 shouldBeRestarted()

```
bool Arcade::Display::Libcaca::shouldBeRestarted ( ) const [final], [virtual]
```

Checks whether you need to restart the current game.

##### Returns

true Restart the game  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.16 shouldExit()

```
bool Arcade::Display::Libcaca::shouldExit ( ) const [final], [virtual]
```

Checks whether you need to exit the program.

##### Returns

true Exit the program  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.17 shouldGoToMenu()

```
bool Arcade::Display::Libcaca::shouldGoToMenu ( ) const [final], [virtual]
```

Checks whether you need to go back to the menu.

##### Returns

true Go back to menu  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.18 switchToNextGame()

```
bool Arcade::Display::Libcaca::switchToNextGame ( ) const [final], [virtual]
```

Checks whether you need to change the current game library.

##### Returns

true Switch to next available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.19 switchToNextLib()

```
bool Arcade::Display::Libcaca::switchToNextLib ( ) const [final], [virtual]
```

Checks whether you need to change the current display library.

##### Returns

true Switch to next available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.20 switchToPreviousGame()

```
bool Arcade::Display::Libcaca::switchToPreviousGame ( ) const [final], [virtual]
```

Checks whether you need to change the current game library.

##### Returns

true Switch to previous available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.10.1.21 switchToPreviousLib()

```
bool Arcade::Display::Libcaca::switchToPreviousLib ( ) const [final], [virtual]
```

Checks whether you need to change the current display library.

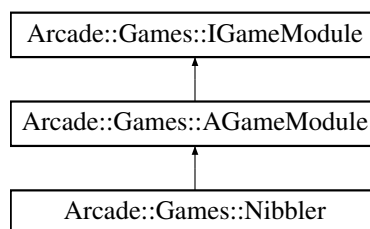
##### Returns

true Switch to previous available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

## 6.11 Arcade::Games::Nibbler Class Reference

Inheritance diagram for Arcade::Games::Nibbler:



### Public Member Functions

- void [reset](#) ( ) final  
*Resets and restarts the game.*
- void [update](#) (const [Arcade::Display::IDisplayModule](#) &displayModule) final  
*Updates the game.*
- void [render](#) ([Arcade::Display::IDisplayModule](#) &displayModule) const final

## Additional Inherited Members

### 6.11.1 Member Function Documentation

#### 6.11.1.1 render()

```
void Arcade::Games::Nibbler::render (
    Arcade::Display::IDisplayModule & lib ) const [final], [virtual]
```

Default game implementation (out of order)

Reimplemented from [Arcade::Games::AGameModule](#).

#### 6.11.1.2 update()

```
void Arcade::Games::Nibbler::update (
    const Arcade::Display::IDisplayModule & lib ) [final], [virtual]
```

Updates the game.

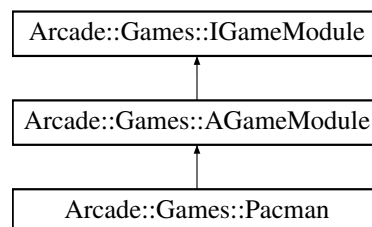
#### Parameters

<i>lib</i>	The display module that will be used to get events that occurred
------------	--

Implements [Arcade::Games::IGameModule](#).

## 6.12 Arcade::Games::Pacman Class Reference

Inheritance diagram for Arcade::Games::Pacman:



### Public Member Functions

- void [reset](#) () final  
*Resets and restarts the game.*
- void [update](#) (const [Arcade::Display::IDisplayModule](#) &lib) final  
*Updates the game.*
- void [render](#) ([Arcade::Display::IDisplayModule](#) &lib) const final

## Additional Inherited Members

### 6.12.1 Member Function Documentation

#### 6.12.1.1 render()

```
void Arcade::Games::Pacman::render (
    Arcade::Display::IDisplayModule & lib ) const [final], [virtual]
```

Default game implementation (out of order)

Reimplemented from [Arcade::Games::AGameModule](#).

#### 6.12.1.2 update()

```
void Arcade::Games::Pacman::update (
    const Arcade::Display::IDisplayModule & lib ) [final], [virtual]
```

Updates the game.

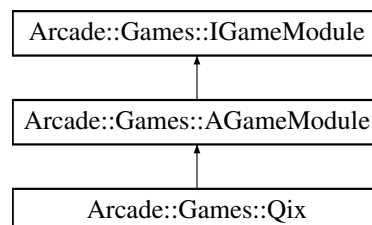
#### Parameters

<i>lib</i>	The display module that will be used to get events that occurred
------------	--

Implements [Arcade::Games::IGameModule](#).

## 6.13 Arcade::Games::Qix Class Reference

Inheritance diagram for Arcade::Games::Qix:



### Public Member Functions

- void [reset](#) () final  
*Resets and restarts the game.*
- void [update](#) (const [Arcade::Display::IDisplayModule](#) &lib) final  
*Updates the game.*

## Additional Inherited Members

### 6.13.1 Member Function Documentation

#### 6.13.1.1 update()

```
void Arcade::Games::Qix::update (
    const Arcade::Display::IDisplayModule & lib ) [final], [virtual]
```

Updates the game.

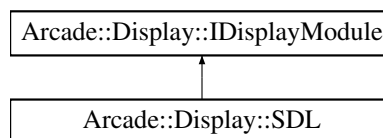
#### Parameters

<i>lib</i>	The display module that will be used to get events that occurred
------------	--

Implements [Arcade::Games::IGameModule](#).

## 6.14 Arcade::Display::SDL Class Reference

Inheritance diagram for Arcade::Display::SDL:



### Public Member Functions

- void [reset](#) () final  
*Resets the library.*
- void [open](#) () final  
*Opens / initializes the window.*
- bool [isOpen](#) () const final  
*Check window status.*
- bool [switchToNextLib](#) () const final  
*Checks whether you need to change the current display library.*
- bool [switchToPreviousLib](#) () const final  
*Checks whether you need to change the current display library.*
- bool [switchToNextGame](#) () const final  
*Checks whether you need to change the current game library.*
- bool [switchToPreviousGame](#) () const final  
*Checks whether you need to change the current game library.*
- bool [shouldBeRestarted](#) () const final

- Checks whether you need to restart the current game.*

  - bool `shouldGoToMenu` () const final

*Checks whether you need to go back to the menu.*
- bool `shouldExit` () const final

*Checks whether you need to exit the program.*
- bool `isKeyPressed` (IDisplayModule::Keys) const final

*Checks whether the current key is being pressed.*
- bool `isKeyPressedOnce` (IDisplayModule::Keys) const final

*Checks whether the current key was pressed during the last frame.*
- float `getDelta` () const final

*Gets the number of frames since last update.*
- void `clear` () const final

*Clears the canvas.*
- void `update` () final

*Runs an update over the events that occurred.*
- void `render` () const final

*Renders the canvas.*
- char `getKeyCode` () const final

*Gets the last pressed character from the keyboard.*
- void `setColor` (IDisplayModule::Colors col) final

*Defines the color of the elements that will be drawn.*
- void `putPixel` (float x, float y) const final

*Displays a pixel.*
- void `putLine` (float x1, float y1, float x2, float y2) const final

*Displays a line.*
- void `putRect` (float x, float y, float w, float h) const final

*Displays a rectangle.*
- void `putFillRect` (float x, float y, float w, float h) const final

*Displays a filled rectangle.*
- void `putCircle` (float x, float y, float rad) const final

*Displays a circle.*
- void `putFillCircle` (float x, float y, float rad) const final

*Displays a filled circle.*
- void `putText` (const std::string &text, unsigned int size, float x, float y) const final

*Displays text.*
- const std::string & `getLibName` () const final

*Gets the library name.*

## Additional Inherited Members

### 6.14.1 Member Function Documentation



#### 6.14.1.1 getDelta()

```
float Arcade::Display::SDL::getDelta ( ) const [final], [virtual]
```

Gets the number of frames since last update.

##### Returns

float Frame count

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.2 getKeyCode()

```
char Arcade::Display::SDL::getKeyCode ( ) const [final], [virtual]
```

Gets the last pressed character from the keyboard.

##### Returns

\0 if nothing was pressed, \b if backspace was pressed, \n if return was pressed, otherwise, a character.

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.3 getLibName()

```
const std::string& Arcade::Display::SDL::getLibName ( ) const [final], [virtual]
```

Gets the library name.

##### Returns

The library's name

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.4 isKeyPressed()

```
bool Arcade::Display::SDL::isKeyPressed (
    IDisplayModule::Keys ) const [final], [virtual]
```

Checks whether the current key is being pressed.

##### Returns

true Key is pressed  
false Key is not pressed

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.5 isKeyPressedOnce()

```
bool Arcade::Display::SDL::isKeyPressedOnce (
    IDisplayModule::Keys ) const [final], [virtual]
```

Checks whether the current key was pressed during the last frame.

##### Returns

true Key is pressed  
false Key is not pressed

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.6 isOpen()

```
bool Arcade::Display::SDL::isOpen ( ) const [final], [virtual]
```

Check window status.

##### Returns

true Window is open  
false Window is closed

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.7 putCircle()

```
void Arcade::Display::SDL::putCircle (
    float x,
    float y,
    float rad ) const [final], [virtual]
```

Displays a circle.

##### Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>rad</i>	Radius of the circle

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.8 putFillCircle()

```
void Arcade::Display::SDL::putFillCircle (
    float x,
    float y,
    float rad ) const [final], [virtual]
```

Displays a filled circle.

##### Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>rad</i>	Radius of the circle

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.9 putFillRect()

```
void Arcade::Display::SDL::putFillRect (
    float x,
    float y,
    float w,
    float h ) const [final], [virtual]
```

Displays a filled rectangle.

##### Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>w</i>	Width of the rectangle
<i>h</i>	Height of the rectangle

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.10 putLine()

```
void Arcade::Display::SDL::putLine (
    float x1,
    float y1,
    float x2,
    float y2 ) const [final], [virtual]
```

Displays a line.

**Parameters**

<i>x1</i>	X coordinates for the first point
<i>y1</i>	Y coordinates for the first point
<i>x2</i>	X coordinates for the second point
<i>y2</i>	Y coordinates for the second point

Implements [Arcade::Display::IDisplayModule](#).

**6.14.1.11 putPixel()**

```
void Arcade::Display::SDL::putPixel (
    float x,
    float y ) const [final], [virtual]
```

Displays a pixel.

**Parameters**

<i>x</i>	X coordinates
<i>y</i>	Y coordinates

Implements [Arcade::Display::IDisplayModule](#).

**6.14.1.12 putRect()**

```
void Arcade::Display::SDL::putRect (
    float x,
    float y,
    float w,
    float h ) const [final], [virtual]
```

Displays a rectangle.

**Parameters**

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>w</i>	Width of the rectangle
<i>h</i>	Height of the rectangle

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.13 putText()

```
void Arcade::Display::SDL::putText (
    const std::string & text,
    unsigned int size,
    float x,
    float y ) const [final], [virtual]
```

Displays text.

##### Parameters

<i>text</i>	The text content
<i>size</i>	The text size
<i>x</i>	X coordinates
<i>y</i>	Y coordinates

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.14 setColor()

```
void Arcade::Display::SDL::setColor (
    IDisplayModule::Colors col ) [final], [virtual]
```

Defines the color of the elements that will be drawn.

##### Parameters

<i>col</i>	The color
------------	-----------

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.15 shouldBeRestarted()

```
bool Arcade::Display::SDL::shouldBeRestarted ( ) const [final], [virtual]
```

Checks whether you need to restart the current game.

##### Returns

true Restart the game  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.16 shouldExit()

```
bool Arcade::Display::SDL::shouldExit ( ) const [final], [virtual]
```

Checks whether you need to exit the program.

##### Returns

true Exit the program  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.17 shouldGoToMenu()

```
bool Arcade::Display::SDL::shouldGoToMenu ( ) const [final], [virtual]
```

Checks whether you need to go back to the menu.

##### Returns

true Go back to menu  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.18 switchToNextGame()

```
bool Arcade::Display::SDL::switchToNextGame ( ) const [final], [virtual]
```

Checks whether you need to change the current game library.

##### Returns

true Switch to next available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.19 switchToNextLib()

```
bool Arcade::Display::SDL::switchToNextLib ( ) const [final], [virtual]
```

Checks whether you need to change the current display library.

##### Returns

true Switch to next available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.20 switchToPreviousGame()

```
bool Arcade::Display::SDL::switchToPreviousGame ( ) const [final], [virtual]
```

Checks whether you need to change the current game library.

##### Returns

true Switch to previous available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

#### 6.14.1.21 switchToPreviousLib()

```
bool Arcade::Display::SDL::switchToPreviousLib ( ) const [final], [virtual]
```

Checks whether you need to change the current display library.

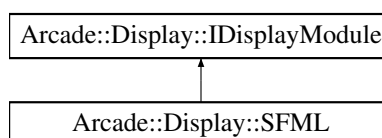
##### Returns

true Switch to previous available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

## 6.15 Arcade::Display::SFML Class Reference

Inheritance diagram for Arcade::Display::SFML:



## Public Member Functions

- void `reset` () final  
*Resets the library.*
- void `open` () final  
*Opens / initializes the window.*
- bool `isOpen` () const final  
*Check window status.*
- bool `switchToNextLib` () const final  
*Checks whether you need to change the current display library.*
- bool `switchToPreviousLib` () const final  
*Checks whether you need to change the current display library.*
- bool `switchToNextGame` () const final  
*Checks whether you need to change the current game library.*
- bool `switchToPreviousGame` () const final  
*Checks whether you need to change the current game library.*
- bool `shouldBeRestarted` () const final  
*Checks whether you need to restart the current game.*
- bool `shouldGoToMenu` () const final  
*Checks whether you need to go back to the menu.*
- bool `shouldExit` () const final  
*Checks whether you need to exit the program.*
- bool `isKeyPressed` (IDisplayModule::Keys) const final  
*Checks whether the current key is being pressed.*
- bool `isKeyPressedOnce` (IDisplayModule::Keys) const final  
*Checks whether the current key was pressed during the last frame.*
- float `getDelta` () const final  
*Gets the number of frames since last update.*
- void `clear` () const final  
*Clears the canvas.*
- void `update` () final  
*Runs an update over the events that occurred.*
- void `render` () const final  
*Renders the canvas.*
- char `getKeyCode` () const final  
*Gets the last pressed character from the keyboard.*
- void `setColor` (IDisplayModule::Colors col) final  
*Defines the color of the elements that will be drawn.*
- void `putPixel` (float x, float y) const final  
*Displays a pixel.*
- void `putLine` (float x1, float y1, float x2, float y2) const final  
*Displays a line.*
- void `putRect` (float x, float y, float w, float h) const final  
*Displays a rectangle.*
- void `putFillRect` (float x, float y, float w, float h) const final  
*Displays a filled rectangle.*
- void `putCircle` (float x, float y, float rad) const final  
*Displays a circle.*
- void `putFillCircle` (float x, float y, float rad) const final  
*Displays a filled circle.*
- void `putText` (const std::string &text, unsigned int size, float x, float y) const final  
*Displays text.*
- const std::string & `getLibName` () const final  
*Gets the library name.*



## Additional Inherited Members

### 6.15.1 Member Function Documentation

#### 6.15.1.1 getDelta()

```
float Arcade::Display::SFML::getDelta ( ) const [final], [virtual]
```

Gets the number of frames since last update.

##### Returns

float Frame count

Implements [Arcade::Display::IDisplayModule](#).

#### 6.15.1.2 getKeyCode()

```
char Arcade::Display::SFML::getKeyCode ( ) const [final], [virtual]
```

Gets the last pressed character from the keyboard.

##### Returns

\0 if nothing was pressed, \b if backspace was pressed, \n if return was pressed, otherwise, a character.

Implements [Arcade::Display::IDisplayModule](#).

#### 6.15.1.3 getLibName()

```
const std::string& Arcade::Display::SFML::getLibName ( ) const [final], [virtual]
```

Gets the library name.

##### Returns

The library's name

Implements [Arcade::Display::IDisplayModule](#).

#### 6.15.1.4 isKeyPressed()

```
bool Arcade::Display::SFML::isKeyPressed (
    IDisplayModule::Keys ) const [final], [virtual]
```

Checks whether the current key is being pressed.

##### Returns

true Key is pressed  
false Key is not pressed

Implements [Arcade::Display::IDisplayModule](#).

#### 6.15.1.5 isKeyPressedOnce()

```
bool Arcade::Display::SFML::isKeyPressedOnce (
    IDisplayModule::Keys ) const [final], [virtual]
```

Checks whether the current key was pressed during the last frame.

##### Returns

true Key is pressed  
false Key is not pressed

Implements [Arcade::Display::IDisplayModule](#).

#### 6.15.1.6 isOpen()

```
bool Arcade::Display::SFML::isOpen ( ) const [final], [virtual]
```

Check window status.

##### Returns

true Window is open  
false Window is closed

Implements [Arcade::Display::IDisplayModule](#).

#### 6.15.1.7 putCircle()

```
void Arcade::Display::SFML::putCircle (
    float x,
    float y,
    float rad ) const [final], [virtual]
```

Displays a circle.

## Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>rad</i>	Radius of the circle

Implements [Arcade::Display::IDisplayModule](#).

**6.15.1.8 putFillCircle()**

```
void Arcade::Display::SFML::putFillCircle (
    float x,
    float y,
    float rad ) const [final], [virtual]
```

Displays a filled circle.

## Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>rad</i>	Radius of the circle

Implements [Arcade::Display::IDisplayModule](#).

**6.15.1.9 putFillRect()**

```
void Arcade::Display::SFML::putFillRect (
    float x,
    float y,
    float w,
    float h ) const [final], [virtual]
```

Displays a filled rectangle.

## Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates
<i>w</i>	Width of the rectangle
<i>h</i>	Height of the rectangle

Implements [Arcade::Display::IDisplayModule](#).

#### 6.15.1.10 putLine()

```
void Arcade::Display::SFML::putLine (
    float x1,
    float y1,
    float x2,
    float y2 ) const [final], [virtual]
```

Displays a line.

##### Parameters

<i>x1</i>	X coordinates for the first point
<i>y1</i>	Y coordinates for the first point
<i>x2</i>	X coordinates for the second point
<i>y2</i>	Y coordinates for the second point

Implements [Arcade::Display::IDisplayModule](#).

#### 6.15.1.11 putPixel()

```
void Arcade::Display::SFML::putPixel (
    float x,
    float y ) const [final], [virtual]
```

Displays a pixel.

##### Parameters

<i>x</i>	X coordinates
<i>y</i>	Y coordinates

Implements [Arcade::Display::IDisplayModule](#).

#### 6.15.1.12 putRect()

```
void Arcade::Display::SFML::putRect (
    float x,
    float y,
    float w,
    float h ) const [final], [virtual]
```

Displays a rectangle.

##### Parameters

<i>x</i>	X coordinates
----------	---------------

## Parameters

<i>y</i>	Y coordinates
<i>w</i>	Width of the rectangle
<i>h</i>	Height of the rectangle

Implements [Arcade::Display::IDisplayModule](#).

## 6.15.1.13 putText()

```
void Arcade::Display::SFML::putText (
    const std::string & text,
    unsigned int size,
    float x,
    float y ) const [final], [virtual]
```

Displays text.

## Parameters

<i>text</i>	The text content
<i>size</i>	The text size
<i>x</i>	X coordinates
<i>y</i>	Y coordinates

Implements [Arcade::Display::IDisplayModule](#).

## 6.15.1.14 setColor()

```
void Arcade::Display::SFML::setColor (
    IDisplayModule::Colors col ) [final], [virtual]
```

Defines the color of the elements that will be drawn.

## Parameters

<i>col</i>	The color
------------	-----------

Implements [Arcade::Display::IDisplayModule](#).

## 6.15.1.15 shouldBeRestarted()

```
bool Arcade::Display::SFML::shouldBeRestarted ( ) const [final], [virtual]
```

Checks whether you need to restart the current game.

**Returns**

true Restart the game  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

**6.15.1.16 shouldExit()**

```
bool Arcade::Display::SFML::shouldExit ( ) const [final], [virtual]
```

Checks whether you need to exit the program.

**Returns**

true Exit the program  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

**6.15.1.17 shouldGoToMenu()**

```
bool Arcade::Display::SFML::shouldGoToMenu ( ) const [final], [virtual]
```

Checks whether you need to go back to the menu.

**Returns**

true Go back to menu  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

**6.15.1.18 switchToNextGame()**

```
bool Arcade::Display::SFML::switchToNextGame ( ) const [final], [virtual]
```

Checks whether you need to change the current game library.

**Returns**

true Switch to next available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

## 6.15.1.19 switchToNextLib()

```
bool Arcade::Display::SFML::switchToNextLib ( ) const [final], [virtual]
```

Checks whether you need to change the current display library.

## Returns

true Switch to next available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

## 6.15.1.20 switchToPreviousGame()

```
bool Arcade::Display::SFML::switchToPreviousGame ( ) const [final], [virtual]
```

Checks whether you need to change the current game library.

## Returns

true Switch to previous available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

## 6.15.1.21 switchToPreviousLib()

```
bool Arcade::Display::SFML::switchToPreviousLib ( ) const [final], [virtual]
```

Checks whether you need to change the current display library.

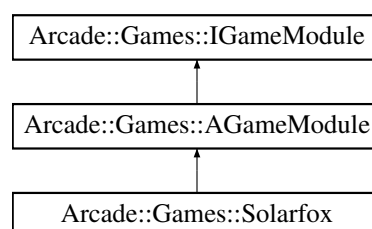
## Returns

true Switch to previous available library  
false Do nothing

Implements [Arcade::Display::IDisplayModule](#).

## 6.16 Arcade::Games::Solarfox Class Reference

Inheritance diagram for Arcade::Games::Solarfox:



## Public Member Functions

- void [reset](#) () final  
*Resets and restarts the game.*
- void [update](#) (const [Arcade::Display::IDisplayModule](#) &lib) final  
*Updates the game.*

## Additional Inherited Members

### 6.16.1 Member Function Documentation

#### 6.16.1.1 [update](#)()

```
void Arcade::Games::Solarfox::update (  
    const Arcade::Display::IDisplayModule & lib ) [final], [virtual]
```

Updates the game.

#### Parameters

<i>lib</i>	The display module that will be used to get events that occurred
------------	--

Implements [Arcade::Games::IGameModule](#).



# Index

## A

- Arcade::Display::IDisplayModule, [23](#)
- addToBestScores
  - Arcade::Games::AGameModule, [12](#)
- AGameModule
  - Arcade::Games::AGameModule, [12](#)
- Arcade, [9](#)
- Arcade::Core, [19](#)
  - Core, [20](#)
- Arcade::Display, [9](#)
- Arcade::Display::IDisplayModule, [20](#)
  - A, [23](#)
  - BLACK, [22](#)
  - BLUE, [22](#)
  - Colors, [22](#)
  - COLORS\_END, [22](#)
  - CYAN, [22](#)
  - D, [23](#)
  - DARK\_GRAY, [22](#)
  - DEFAULT, [22](#)
  - DOWN, [23](#)
  - E, [23](#)
  - ENTER, [23](#)
  - ESCAPE, [23](#)
  - getDelta, [23](#)
  - getKeyCode, [23](#)
  - getLibName, [23](#)
  - GREEN, [22](#)
  - I, [23](#)
  - isKeyPressed, [24](#)
  - isKeyPressedOnce, [24](#)
  - isOpen, [24](#)
  - J, [23](#)
  - K, [23](#)
  - Keys, [22](#)
  - KEYS\_END, [23](#)
  - LEFT, [22](#)
  - LIGHT\_BLUE, [22](#)
  - LIGHT\_CYAN, [22](#)
  - LIGHT\_GRAY, [22](#)
  - LIGHT\_GREEN, [22](#)
  - LIGHT\_MAGENTA, [22](#)
  - LIGHT\_RED, [22](#)
  - LIGHT\_YELLOW, [22](#)
  - M, [23](#)
  - MAGENTA, [22](#)
  - putCircle, [25](#)
  - putFillCircle, [25](#)
  - putFillRect, [26](#)
  - putLine, [26](#)
  - putPixel, [26](#)
  - putRect, [27](#)
  - putText, [27](#)
  - Q, [23](#)
  - R, [23](#)
  - RED, [22](#)
  - RIGHT, [22](#)
  - S, [23](#)
  - setColor, [27](#)
  - shouldBeRestarted, [28](#)
  - shouldExit, [28](#)
  - shouldGoToMenu, [28](#)
  - SPACE, [23](#)
  - switchToNextGame, [29](#)
  - switchToNextLib, [29](#)
  - switchToPreviousGame, [29](#)
  - switchToPreviousLib, [30](#)
  - U, [23](#)
  - UP, [23](#)
  - W, [23](#)
  - WHITE, [22](#)
  - X, [23](#)
  - YELLOW, [22](#)
  - Z, [23](#)
- Arcade::Display::Libcaca, [35](#)
  - getDelta, [37](#)
  - getKeyCode, [37](#)
  - getLibName, [37](#)
  - isKeyPressed, [38](#)
  - isKeyPressedOnce, [38](#)
  - isOpen, [38](#)
  - putCircle, [39](#)
  - putFillCircle, [39](#)
  - putFillRect, [40](#)
  - putLine, [40](#)
  - putPixel, [40](#)
  - putRect, [41](#)
  - putText, [41](#)
  - setColor, [41](#)
  - shouldBeRestarted, [42](#)
  - shouldExit, [42](#)
  - shouldGoToMenu, [42](#)
  - switchToNextGame, [43](#)
  - switchToNextLib, [43](#)
  - switchToPreviousGame, [43](#)
  - switchToPreviousLib, [44](#)
- Arcade::Display::SDL, [47](#)
  - getDelta, [48](#)

- getKeyCode, [49](#)
- getLibName, [49](#)
- isKeyPressed, [49](#)
- isKeyPressedOnce, [49](#)
- isOpen, [50](#)
- putCircle, [50](#)
- putFillCircle, [50](#)
- putFillRect, [51](#)
- putLine, [51](#)
- putPixel, [52](#)
- putRect, [52](#)
- putText, [52](#)
- setColor, [53](#)
- shouldBeRestarted, [53](#)
- shouldExit, [53](#)
- shouldGoToMenu, [54](#)
- switchToNextGame, [54](#)
- switchToNextLib, [54](#)
- switchToPreviousGame, [55](#)
- switchToPreviousLib, [55](#)
- Arcade::Display::SFML, [55](#)
  - getDelta, [57](#)
  - getKeyCode, [57](#)
  - getLibName, [57](#)
  - isKeyPressed, [57](#)
  - isKeyPressedOnce, [58](#)
  - isOpen, [58](#)
  - putCircle, [58](#)
  - putFillCircle, [59](#)
  - putFillRect, [59](#)
  - putLine, [59](#)
  - putPixel, [60](#)
  - putRect, [60](#)
  - putText, [61](#)
  - setColor, [61](#)
  - shouldBeRestarted, [61](#)
  - shouldExit, [62](#)
  - shouldGoToMenu, [62](#)
  - switchToNextGame, [62](#)
  - switchToNextLib, [62](#)
  - switchToPreviousGame, [63](#)
  - switchToPreviousLib, [63](#)
- Arcade::Exceptions, [10](#)
- Arcade::Exceptions::ArcadeException, [15](#)
  - ArcadeException, [16](#)
  - getComponent, [16](#)
- Arcade::Exceptions::BadFileException, [17](#)
  - BadFileException, [17](#)
- Arcade::Exceptions::BadInstanciationException, [18](#)
  - BadInstanciationException, [18](#)
- Arcade::Exceptions::InvalidLibraryException, [34](#)
  - InvalidLibraryException, [35](#)
- Arcade::Games, [10](#)
- Arcade::Games::AGameModule, [11](#)
  - addToBestScores, [12](#)
  - AGameModule, [12](#)
  - getBestScores, [13](#)
  - getLibName, [13](#)
  - getScore, [13](#)
  - loadFromFile, [13, 14](#)
  - render, [14](#)
  - saveToFile, [14, 15](#)
  - setPlayerName, [15](#)
- Arcade::Games::Centipede, [18](#)
  - update, [19](#)
- Arcade::Games::IGameModule, [30](#)
  - getBestScores, [31](#)
  - getLibName, [31](#)
  - getScore, [32](#)
  - loadFromFile, [32](#)
  - render, [33](#)
  - saveToFile, [33](#)
  - setPlayerName, [34](#)
  - update, [34](#)
- Arcade::Games::Nibbler, [44](#)
  - render, [45](#)
  - update, [45](#)
- Arcade::Games::Pacman, [45](#)
  - render, [46](#)
  - update, [46](#)
- Arcade::Games::Qix, [46](#)
  - update, [47](#)
- Arcade::Games::Solarfox, [63](#)
  - update, [64](#)
- ArcadeException
  - Arcade::Exceptions::ArcadeException, [16](#)
- BadFileException
  - Arcade::Exceptions::BadFileException, [17](#)
- BadInstanciationException
  - Arcade::Exceptions::BadInstanciationException, [18](#)
- BLACK
  - Arcade::Display::IDisplayModule, [22](#)
- BLUE
  - Arcade::Display::IDisplayModule, [22](#)
- Colors
  - Arcade::Display::IDisplayModule, [22](#)
- COLORS\_END
  - Arcade::Display::IDisplayModule, [22](#)
- Core
  - Arcade::Core, [20](#)
- CYAN
  - Arcade::Display::IDisplayModule, [22](#)
- D
  - Arcade::Display::IDisplayModule, [23](#)
- DARK\_GRAY
  - Arcade::Display::IDisplayModule, [22](#)
- DEFAULT
  - Arcade::Display::IDisplayModule, [22](#)
- DOWN
  - Arcade::Display::IDisplayModule, [23](#)
- E
  - Arcade::Display::IDisplayModule, [23](#)

- ENTER
  - Arcade::Display::IDisplayModule, [23](#)
- ESCAPE
  - Arcade::Display::IDisplayModule, [23](#)
- getBestScores
  - Arcade::Games::AGameModule, [13](#)
  - Arcade::Games::IGameModule, [31](#)
- getComponent
  - Arcade::Exceptions::ArcadeException, [16](#)
- getDelta
  - Arcade::Display::IDisplayModule, [23](#)
  - Arcade::Display::Libcaca, [37](#)
  - Arcade::Display::SDL, [48](#)
  - Arcade::Display::SFML, [57](#)
- getKeyCode
  - Arcade::Display::IDisplayModule, [23](#)
  - Arcade::Display::Libcaca, [37](#)
  - Arcade::Display::SDL, [49](#)
  - Arcade::Display::SFML, [57](#)
- getLibName
  - Arcade::Display::IDisplayModule, [23](#)
  - Arcade::Display::Libcaca, [37](#)
  - Arcade::Display::SDL, [49](#)
  - Arcade::Display::SFML, [57](#)
  - Arcade::Games::AGameModule, [13](#)
  - Arcade::Games::IGameModule, [31](#)
- getScore
  - Arcade::Games::AGameModule, [13](#)
  - Arcade::Games::IGameModule, [32](#)
- GREEN
  - Arcade::Display::IDisplayModule, [22](#)
- I
  - Arcade::Display::IDisplayModule, [23](#)
- InvalidLibraryException
  - Arcade::Exceptions::InvalidLibraryException, [35](#)
- isKeyPressed
  - Arcade::Display::IDisplayModule, [24](#)
  - Arcade::Display::Libcaca, [38](#)
  - Arcade::Display::SDL, [49](#)
  - Arcade::Display::SFML, [57](#)
- isKeyPressedOnce
  - Arcade::Display::IDisplayModule, [24](#)
  - Arcade::Display::Libcaca, [38](#)
  - Arcade::Display::SDL, [49](#)
  - Arcade::Display::SFML, [58](#)
- isOpen
  - Arcade::Display::IDisplayModule, [24](#)
  - Arcade::Display::Libcaca, [38](#)
  - Arcade::Display::SDL, [50](#)
  - Arcade::Display::SFML, [58](#)
- J
  - Arcade::Display::IDisplayModule, [23](#)
- K
  - Arcade::Display::IDisplayModule, [23](#)
- Keys
  - Arcade::Display::IDisplayModule, [22](#)
- KEYS\_END
  - Arcade::Display::IDisplayModule, [23](#)
- LEFT
  - Arcade::Display::IDisplayModule, [22](#)
- LIGHT\_BLUE
  - Arcade::Display::IDisplayModule, [22](#)
- LIGHT\_CYAN
  - Arcade::Display::IDisplayModule, [22](#)
- LIGHT\_GRAY
  - Arcade::Display::IDisplayModule, [22](#)
- LIGHT\_GREEN
  - Arcade::Display::IDisplayModule, [22](#)
- LIGHT\_MAGENTA
  - Arcade::Display::IDisplayModule, [22](#)
- LIGHT\_RED
  - Arcade::Display::IDisplayModule, [22](#)
- LIGHT\_YELLOW
  - Arcade::Display::IDisplayModule, [22](#)
- loadFromFile
  - Arcade::Games::AGameModule, [13](#), [14](#)
  - Arcade::Games::IGameModule, [32](#)
- M
  - Arcade::Display::IDisplayModule, [23](#)
- MAGENTA
  - Arcade::Display::IDisplayModule, [22](#)
- putCircle
  - Arcade::Display::IDisplayModule, [25](#)
  - Arcade::Display::Libcaca, [39](#)
  - Arcade::Display::SDL, [50](#)
  - Arcade::Display::SFML, [58](#)
- putFillCircle
  - Arcade::Display::IDisplayModule, [25](#)
  - Arcade::Display::Libcaca, [39](#)
  - Arcade::Display::SDL, [50](#)
  - Arcade::Display::SFML, [59](#)
- putFillRect
  - Arcade::Display::IDisplayModule, [26](#)
  - Arcade::Display::Libcaca, [40](#)
  - Arcade::Display::SDL, [51](#)
  - Arcade::Display::SFML, [59](#)
- putLine
  - Arcade::Display::IDisplayModule, [26](#)
  - Arcade::Display::Libcaca, [40](#)
  - Arcade::Display::SDL, [51](#)
  - Arcade::Display::SFML, [59](#)
- putPixel
  - Arcade::Display::IDisplayModule, [26](#)
  - Arcade::Display::Libcaca, [40](#)
  - Arcade::Display::SDL, [52](#)
  - Arcade::Display::SFML, [60](#)
- putRect
  - Arcade::Display::IDisplayModule, [27](#)
  - Arcade::Display::Libcaca, [41](#)
  - Arcade::Display::SDL, [52](#)
  - Arcade::Display::SFML, [60](#)

- putText
  - Arcade::Display::IDisplayModule, 27
  - Arcade::Display::Libcaca, 41
  - Arcade::Display::SDL, 52
  - Arcade::Display::SFML, 61
- Q
  - Arcade::Display::IDisplayModule, 23
- R
  - Arcade::Display::IDisplayModule, 23
- RED
  - Arcade::Display::IDisplayModule, 22
- render
  - Arcade::Games::AGameModule, 14
  - Arcade::Games::IGameModule, 33
  - Arcade::Games::Nibbler, 45
  - Arcade::Games::Pacman, 46
- RIGHT
  - Arcade::Display::IDisplayModule, 22
- S
  - Arcade::Display::IDisplayModule, 23
- saveToFile
  - Arcade::Games::AGameModule, 14, 15
  - Arcade::Games::IGameModule, 33
- setColor
  - Arcade::Display::IDisplayModule, 27
  - Arcade::Display::Libcaca, 41
  - Arcade::Display::SDL, 53
  - Arcade::Display::SFML, 61
- setPlayerName
  - Arcade::Games::AGameModule, 15
  - Arcade::Games::IGameModule, 34
- shouldBeRestarted
  - Arcade::Display::IDisplayModule, 28
  - Arcade::Display::Libcaca, 42
  - Arcade::Display::SDL, 53
  - Arcade::Display::SFML, 61
- shouldExit
  - Arcade::Display::IDisplayModule, 28
  - Arcade::Display::Libcaca, 42
  - Arcade::Display::SDL, 53
  - Arcade::Display::SFML, 62
- shouldGoToMenu
  - Arcade::Display::IDisplayModule, 28
  - Arcade::Display::Libcaca, 42
  - Arcade::Display::SDL, 54
  - Arcade::Display::SFML, 62
- SPACE
  - Arcade::Display::IDisplayModule, 23
- switchToNextGame
  - Arcade::Display::IDisplayModule, 29
  - Arcade::Display::Libcaca, 43
  - Arcade::Display::SDL, 54
  - Arcade::Display::SFML, 62
- switchToNextLib
  - Arcade::Display::IDisplayModule, 29
  - Arcade::Display::Libcaca, 43
- Arcade::Display::SDL, 54
- Arcade::Display::SFML, 62
- switchToPreviousGame
  - Arcade::Display::IDisplayModule, 29
  - Arcade::Display::Libcaca, 43
  - Arcade::Display::SDL, 55
  - Arcade::Display::SFML, 63
- switchToPreviousLib
  - Arcade::Display::IDisplayModule, 30
  - Arcade::Display::Libcaca, 44
  - Arcade::Display::SDL, 55
  - Arcade::Display::SFML, 63
- U
  - Arcade::Display::IDisplayModule, 23
- UP
  - Arcade::Display::IDisplayModule, 23
- update
  - Arcade::Games::Centipede, 19
  - Arcade::Games::IGameModule, 34
  - Arcade::Games::Nibbler, 45
  - Arcade::Games::Pacman, 46
  - Arcade::Games::Qix, 47
  - Arcade::Games::Solarfox, 64
- W
  - Arcade::Display::IDisplayModule, 23
- WHITE
  - Arcade::Display::IDisplayModule, 22
- X
  - Arcade::Display::IDisplayModule, 23
- YELLOW
  - Arcade::Display::IDisplayModule, 22
- Z
  - Arcade::Display::IDisplayModule, 23