

codeconventions of blobby volley 2

the blobby volley developers

October 31, 2013

1 Structure of Files

This chapter gives an overview about the used .cpp and .h file structure convention used in the Blobby Volley 2 project. If you want to contribute some code to the project it's important you take care of it.

1.1 Headerfile

Every headerfile of Blobby Volley 2 needs in top of the page the license and author text shown in listing 1.

Listing 1: License and author text of a headerfile

```
/*=====
Blobby Volley 2
Copyright (C) 2006 Jonathan Sieber (jonathan_sieber@yahoo.de)
Copyright (C) 2006 Daniel Knobe (daniel-knobe@web.de)

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
=====*/
```

After the header follows an empty line, followed by a brief filedescription which must look like in listing 2.

Listing 2: Filedescription of an headerfile

```
/**
 * @file Filename.h
 * @brief The short description of the file
 */
```

1.2 Bodyfile

```
/*=====
Blobby Volley 2
Copyright (C) 2006 Jonathan Sieber (jonathan_sieber@yahoo.de)
Copyright (C) 2006 Daniel Knobe (daniel-knobe@web.de)

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
```

```
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
```

```
*/
```

```
/* header include */
```

```
/* includes */
```

```
/* implementation */
```

1.3 Includes

Includes are to be sorted by library. They should occur in the following order:

- Stdlib
- Boost
- Other 3rd party libraries we use (for Example: lua, tinyxml, raknet, SDL)
- Blobby

1.4 Coding Style

This section describes the coding style. If you think something is missing just add it.

1.4.1 Common

- Opening curly braces are placed in a new line
- Closing curly braces are followed by an empty line
- The left indentation should be done by tabs, the rest by spaces
- Do not write more than one statement into one source line
- Break statements within switch are followed by an empty line
- Use smart pointers when ever possible
- Code should be gcc 4.6 compatible
- Code should compile in c++11 mode
- Don't use the deprecated exception handling of older c++ versions

- Binary/ternary operators are separated from their operands by spaces (exception: long expression - for grouping purposes)
- Leaving out braces for one line code blocks after if/for is allowed when the following instruction takes only one line, but must be followed by an empty line
- Use c++ headers instead of c ones for stdlib includes (e.g. `<cmath>` instead of `<math.h>`)
- Use c++ strings instead of c strings if possible
- Prefer references over pointers

1.4.2 Classes

- Class names start with an upper case letter
- Member variables are to be prefixed with `m`
- Member functions start with a lower case letter
- Members occur in the following order: public, protected, private
- Access level specifiers (is it called that way? / public, protected, private) are indented one level
- Members are indented two levels
- For big classes, it might be a good idea to group member functions into different categories
- Initialize members by constructors initphase whenever possible

1.5 Namespaces

TODO

2 Things you should not do

Every contribution with this attributes will be rejected.

2.1 Pattern

The following patterns are not allowed to use:

- Singleton (http://en.wikipedia.org/wiki/Singleton_pattern)

3 Stuff that must be integrated in this file correctly

TODO: - Write a more detailed code-convention document - Detailed description of every section needed

Following files implement the convention: - RenderManagerSDL.cpp - RenderManagerGL2D.cpp - UserConfig.cpp - TextManager.cpp - SpeedController.cpp - SoundManager.cpp - ScriptedInputSource.cpp - ReplayRecorder.cpp - ReplayPlayer.cpp - RenderManagerGP2X.cpp - RenderManager.cpp - Player.cpp - PhysicWorld.cpp - NetworkMessage.cpp
TODO: - all .cpp in subfolder - src folder a - NetworkGame.cpp - ReplayLoader.cpp