Contribution Guidelines

=======================

In order to contribute to mGBA, there are a few things to be mindful of so as to ease the process.

Filing issues

-------------

New issues should be filed on the [mGBA GitHub Issues tracker](http://mgba.io/i/). When filing issues, please include the following information:

\* The build you are using. For recent builds, this is visible in the title bar. For example, `0.3-2134-4ec19aa`. On older builds, such as 0.2.1, this is not present, so please specify the version you downloaded or built. If present, this contains the version, branch name (if not `master`), a revision number and a truncated revision hash. For this example, it means that it's version 0.3, on `master`, commit number 2134 and revision hash `4ec19aa`. Additionally, `-dirty` will be appended if there are local changes that haven't been commited.

\* The operating system you're using, for example Windows 7 32-bit or Ubuntu 15.04 64-bit.

\* Your CPU and graphics card (usually not necessary). For example, Core i5-3570K and AMD Radeon R9 280X.

Please also describe the issue in as much detail as possible, including the name of the games you have reproduced the issue on, and how you managed to enter the buggy state. If applicable, savestates can be renamed to be .png files and attached to the issue directly.

Filing pull requests

--------------------

When filing a pull request, please make sure you adhere to the coding style as outlined below, and are aware of the requirements for licensing. Furthermore, please make sure all commits in the pull request have coherent commit messages as well as the name of the component being modified in the commit message.

Some components are as follows:

\* ARM7: The ARM core

\* GBA: GBA code

\* GBA Memory: Memory-specific

\* GBA Video: Video, rendering

\* GBA Audio: Audio processing

\* GBA SIO: Serial I/O, multiplayer, link

\* GBA Hardware: Extra devices, e.g. gyro, light sensor

\* GBA RR: Rerecording features

\* GBA Thread: Thread-layer abstractions

\* GBA BIOS: High-level BIOS

\* Qt: Qt port-related code

\* SDL: SDL port-related code (including as used in other ports)

\* Video: Video recording code

\* Util: Common utility code

\* Tools: Miscellaneous tools

\* Debugger: Included debugging functionality

\* All: Changes that don't touch specific components but affect the project overall

Coding Style

------------

mGBA aims to have a consistent, clean codebase, so when contributing code to mGBA, please adhere to the following rules. If a pull request has style errors, you will be asked to fix them before the PR will be accepted.

### Naming

Variable names, including parameters, should all be in camelCase. File-scoped static variables must start with an underscore.

C struct names should start with a capital letter, and functions relating to these structs should start with the name of the class (including the capital letter) and be in camelCase after. C struct should not be `typedef`ed.

Functions not associated with structs should be in camelCase throughout. Static functions not associated with structs must start with an underscore.

Enum values and `#define`s should be all caps with underscores.

Good:

static int \_localVariable;

struct LocalStruct {

void (\*methodName)(struct LocalStruct struct, param);

int memberName;

};

enum {

ENUM\_ITEM\_1,

ENUM\_ITEM\_2

};

void LocalStructCreate(struct LocalStruct\* struct);

void functionName(int argument);

static void \_LocalStructUse(struct LocalStruct\* struct);

static void \_function2(int argument2);

C++ classes should be confined to namespaces. For the Qt port, this namespace is called `QGBA`.

Class names should be handled similarly to C structs. Fields should be prefixed according to their scoping:

\* `m\_` for non-static member.

\* `s\_` for static member.

### Braces

Braces do not go on their own lines, apart from the terminating brace. There should be a single space between the condition clause and the brace. Furthermore, braces must be used even for single-line blocks.

Good:

if (condition) {

block;

} else if (condition2) {

block2;

} else {

block3;

}

Bad (separate line):

if (condition)

{

block;

}

else if (condition2)

{

block2;

}

else

{

block3;

}

Bad (missing braces):

if (condition)

statement;

else if (condition2)

statement2;

else

statement3;

Bad (missing space):

if (condition){

block;

}

### Spacing

Indentation should be done using tabs and should match the level of braces. Alignment within a line should be done sparingly, but only done with spaces.

### Header guards

For C headers guards, the define should be the filename (including H), all-caps, with underscores instead of punctuation.

Good:

#ifndef FILE\_NAME\_H

#define FILE\_NAME\_H

// Header

#endif

There should be no comment on the `#endif`.

For Qt (C++ header guards), the define should start with `QGBA\_` and not include `\_H`, but is otherwise the same. This is mostly for legacy reasons., and may change in the future.

Good:

#ifndef QGBA\_FILE\_NAME

#define QGBA\_FILE\_NAME

// Header

#endif

### Other

Block statements such as `if`, `while` and `for` should have a space between the type of block and the parenthesis.

Good:

while (condition) {

block;

}

Bad:

while(condition) {

block;

}

In C code, use `0` instead of `NULL`. This is mostly for legacy reasons and may change in the future. C code should also use `bool` types and values `true` and `false` instead of `1` and `0` where applicable. In C++ code, use `nullptr` instead of `NULL` or `0`.

If a statement has no body, putting braces is not required, and a semicolon can be used. This is not required, but is suggested.

Good:

while (f());

Bad:

while (f()) {}

For infinite loops that `break` statements internally, `while (true)` is preferred over `for (;;)`.

Licensing

---------

mGBA is licensed under the [Mozilla Public License version 2.0](https://www.mozilla.org/MPL/2.0/). This entails a few things when it comes to adding code to mGBA.

\* New code to mGBA will be licensed under the MPL 2.0 license.

\* GPL-licensed code cannot be added to mGBA upstream, but can be linked with mGBA when compiled.

\* MIT, BSD, CC0, etc., code can be added to mGBA upstream, but preferably in the `third-party` section if applicable.