# How to contribute

mruby is an open-source project which is looking forward to each contribution.

## Your Pull Request

To make it easy to review and understand your change please keep the following

things in mind before submitting your pull request:

\* Work on the latest possible state of \*\*mruby/master\*\*

\* Create a branch which is dedicated to your change

\* Test your changes before creating a pull request (```rake test```)

\* If possible write a test case which confirms your change

\* Don't mix several features or bug-fixes in one pull request

\* Create a meaningful commit message

\* Explain your change (i.e. with a link to the issue you are fixing)

\* Use mrbgem to provide non ISO features (classes, modules and methods) unless

you have a special reason to implement them in the core

## Coding conventions

How to style your C and Ruby code which you want to submit.

### C code

The core part (parser, bytecode-interpreter, core-lib, etc.) of mruby is

written in the C programming language. Please note the following hints for your

C code:

#### Comply with C99 (ISO/IEC 9899:1999)

mruby should be highly portable to other systems and compilers. For this it is

recommended to keep your code as close as possible to the C99 standard

(http://www.open-std.org/jtc1/sc22/WG14/www/docs/n1256.pdf).

Although we target C99, we've heard some compilers in the embedded environment

still requires declarations of local variables to be at the beginning of a

scope. Until we confirm the situation has changed, we use the old-style

variable declaration.

Visual C++ is also an important target for mruby (supported version is 2013 or

later). For this reason features that are not supported by Visual C++ may not

be used (e.g. `%z` of `strftime()`).

NOTE: Old GCC requires `-std=gnu99` option to enable C99 support.

#### Reduce library dependencies to a minimum

The dependencies to libraries should be kept to an absolute minimum. This

increases the portability but makes it also easier to cut away parts of mruby

on-demand.

#### Don't use C++ style comments

/\* This is the preferred comment style \*/

Use C++ style comments only for temporary comment e.g. commenting out some code lines.

#### Insert a break after the method return value:

int

main(void)

{

...

}

### Ruby code

Parts of the standard library of mruby are written in the Ruby programming

language itself. Please note the following hints for your Ruby code:

#### Comply with the Ruby standard (ISO/IEC 30170:2012)

mruby is currently targeting to execute Ruby code which complies to ISO/IEC

30170:2012 (http://www.iso.org/iso/iso\_catalogue/catalogue\_tc/catalogue\_detail.htm?csnumber=59579).