Contributing to R Core

R Forwards

(updated on 2020-07-15)

Contents

9

1 R Core Developer's Guide 11
1.1 Quick Reference . . . 11
1.2 Quick Links . . . 11
1.3 Status of R Core Branches . . . 11
1.4 Contributing . . . 11
1.5 Proposing Changes to R Core . . . 11
1.6 Other Interpreter Implementations . . . 11
1.7 Key Resources . . . 11
1.8 Additional Resources . . . 11
1.9 Code of Conduct . . . 11

2 Getting Started 13
2.1 Install git . . . 13
2.2 Get the source code . . . 13
2.3 Compile and build . . . 13
2.4 Install dependencies . . . 13
2.5 Regenerate `configure` . . . 13
2.6 Troubleshoot the build . . . 13
2.7 Editors and Tools . . . 13
2.8 Directory structure . . . 13

3 Where to Get Help 15
3.1 Ask r-dev . . . 15
3.2 Zulip . . . 15
3.3 Core Mentorship . . . 15
3.4 Core Developers Office Hours . . . 15
3.5 Mailing Lists . . . 15
3.6 File a Bug . . . 15

4 Lifecycle of a Pull Request 17
4.1 Introduction . . . 18
4.2 Quick Guide . . . 18

4.3 Step-by-step Guide 18
4.4 Making Good PRs 18
4.5 Making Good Commits 18
4.6 Licensing 18
4.7 Submitting 18
4.8 Converting an Existing Patch 18
4.9 Reviewing 18
4.10 Leaving a Pull Request Review 18
4.11 Committing/Rejecting 18
4.12 Crediting 18

5 Lifecycle of a Pull Request **19**
5.1 Introduction 20
5.2 Quick Guide 20
5.3 Step-by-step Guide 20
5.4 Making Good PRs 20
5.5 Making Good Commits 20
5.6 Licensing 20
5.7 Submitting 20
5.8 Converting an Existing Patch 20
5.9 Reviewing 20
5.10 Leaving a Pull Request Review 20
5.11 Committing/Rejecting 20
5.12 Crediting 20

6 Help **21**
6.1 Identify specific areas where help is needed 21

7 Running & Writing Tests **23**
7.1 Running 23
7.2 Writing 23
7.3 Benchmarks 23

8 Increase Test Coverage **25**
8.1 Common Gotchas 25
8.2 Measuring Coverage 25
8.3 Filing the Issue 25
8.4 Measuring coverage of C code 25

9 Helping with Documentation **27**
9.1 R Core Documentation 27
9.2 Helping with Documentation 27
9.3 Proofreading 27
9.4 Helping with the Developer's Guide 27
9.5 Developer's Guide workflow 27

10 Documenting R **29**

10.1 Introduction 30
10.2 Style Guide 30
10.3 reStructuredText Primer 30
10.4 Additional Markup Constructs 30
10.5 Building the documentation 30
10.6 Translating 30

11 Silence Warnings From the Test Suite 31

12 Silence Warnings From the Test Suite 33

13 Fixing "easy" Issues (and Beyond) 35

14 Issue Tracking 37
14.1 Using the Issue Tracker 37
14.2 Disagreement With a Resolution on the Issue Tracker 37
14.3 Helping Triage Issues 37
14.4 Gaining the "Developer" Role on the Issue Tracker 37
14.5 The Meta Tracker 37

15 Triaging an Issue 39
15.1 Python triage team 40
15.2 Becoming a member of the Python triage team 40
15.3 Fields in the Issue Tracker 40
15.4 Generating Special Links in a Comment 40
15.5 Checklist for Triaging 40

16 Following R's Development 41
16.1 Mailing Lists 41
16.2 Zulip 41
16.3 IRC 41
16.4 Blogs 41
16.5 Standards of behaviour in these communication channels 41
16.6 Setting Expectations for Open Source Participation 41
16.7 Additional Repositories 41

17 Porting R to a new platform 43

18 How to Become a Core Developer 45
18.1 What it Takes 45
18.2 What it Means 45
18.3 Gaining Commit Privileges 45
18.4 Responsibilities 45

19 Developer Log 47
19.1 Procedure for Granting or Dropping Access 47

20 Accepting Pull Requests **49**
20.1 Is the PR ready to be accepted 49
20.2 Handling Others' Code 49
20.3 Contributor Licensing Agreements 49
20.4 Checking if the CLA has been received 49
20.5 What's New and News Entries 49
20.6 Working with Git 49

21 Development Cycle **51**
21.1 Branches 52
21.2 Stages 52
21.3 Repository Administration 52

22 Continuous Integration **53**
22.1 Checking results of automatic builds 53
22.2 Stability 53
22.3 Flags-dependent failures 53
22.4 Ordering-dependent failures 53
22.5 Transient failures 53
22.6 Custom builders 53

23 Adding to the Stdlib **55**
23.1 Adding to a pre-existing module 55
23.2 Adding a new module 55

24 Experts Index **57**
24.1 Stdlib 57
24.2 Tools 57
24.3 Platforms 57
24.4 Miscellaneous 57
24.5 Documentation Translations 57

25 gdb Support **59**
25.1 gdb 7 and later 59
25.2 gdb 6 and earlier 59
25.3 Updating auto-load-safe-path to allow test_gdb to run 59

26 Exploring R Internals **61**

27 Changing R's Grammar **63**

28 Design of R Compiler **65**

29 Design of R's Garbage Collector **67**

30 Updating standard library extension modules **69**

31 Coverity Scan **71**

32 Dynamic Analysis with Clang **73**

33 Running a buildbot worker **75**

34 Core Developer Motivations and Affiliations **77**
34.1 Published entries 77
34.2 Goals of this page 77
34.3 Limitations of Scope 77

35 Cheatsheets **79**

36 Accepting Pull Requests **81**
36.1 Is the PR ready to be accepted 81
36.2 Handling Others' Code 81
36.3 Contributor Licensing Agreements 81
36.4 Checking if the CLA has been received 81
36.5 What's New and News Entries 81
36.6 Working with Git 81

Chapter 1

R Core Developer's Guide

This guide is heavily influenced by the Python Developer Guide, and is a comprehensive resource for contributing to R Core – for both new and experienced contributors. It is maintained by [XXX]. We welcome your contributions to R Core!

This box denotes a tip for the reader.

1.1 Quick Reference

1.2 Quick Links

1.3 Status of R Core Branches

1.4 Contributing

1.5 Proposing Changes to R Core

1.6 Other Interpreter Implementations

1.7 Key Resources

1.8 Additional Resources

1.9 Code of Conduct

Chapter 2

Getting Started

2.1 Install git

2.2 Get the source code

2.3 Compile and build

2.3.1 UNIX

2.3.2 Windows

2.4 Install dependencies

2.4.1 Linux

2.4.2 maxOS and OS X

2.5 Regenerate `configure`

2.6 Troubleshoot the build

2.6.1 Avoid recreating auto-generated files

2.7 Editors and Tools

2.8 Directory structure

Chapter 3

Where to Get Help

3.1 Ask r-dev

3.2 Zulip

3.3 Core Mentorship

3.4 Core Developers Office Hours

3.5 Mailing Lists

3.6 File a Bug

Chapter 4

Lifecycle of a Pull Request

4.1 Introduction

4.2 Quick Guide

4.3 Step-by-step Guide

4.3.1 Resolving Merge Confilcts

4.4 Making Good PRs

4.5 Making Good Commits

4.6 Licensing

4.7 Submitting

4.8 Converting an Existing Patch

4.9 Reviewing

4.9.1 How to Review a Pull Request

4.10 Leaving a Pull Request Review

4.11 Committing/Rejecting

4.12 Crediting

Chapter 5

Lifecycle of a Pull Request

5.1 Introduction

5.2 Quick Guide

5.3 Step-by-step Guide

5.3.1 Resolving Merge Confilcts

5.4 Making Good PRs

5.5 Making Good Commits

5.6 Licensing

5.7 Submitting

5.8 Converting an Existing Patch

5.9 Reviewing

5.9.1 How to Review a Pull Request

5.10 Leaving a Pull Request Review

5.11 Committing/Rejecting

5.12 Crediting

Chapter 6

Help

6.1 Identify specific areas where help is needed

- Help needed on recommended packages: MASS, survival
- Testing pre-releases
  - write how-to?
  - set up virtual machines for people to test on?
- Responding on R-devel/R-package-devel
  - introduce moderation?

Chapter 7

Running & Writing Tests

7.1 Running

7.1.1 Unexpected Skips

7.2 Writing

7.3 Benchmarks

Chapter 8

Increase Test Coverage

8.1 Common Gotchas

8.2 Measuring Coverage

8.2.1 Using `covr`

8.3 Filing the Issue

8.4 Measuring coverage of C code

Chapter 9

Helping with Documentation

9.1 R Core Documentation

9.2 Helping with Documentation

9.3 Proofreading

9.4 Helping with the Developer's Guide

9.5 Developer's Guide workflow

Chapter 10

Documenting R

10.1 Introduction

10.2 Style Guide

10.2.1 Whitespace

10.2.2 Footnotes

10.2.3 Capitalization

10.2.4 Affirmative Tone

10.2.5 Economy of Expression

10.2.6 Security Consideration

10.2.7 Code Examples

10.2.8 Code Equivalents

10.2.9 Audience

10.3 reStructuredText Primer

10.3.1 Paragraphs

10.3.2 Inline markup

10.3.3 Lists and Quotes

10.3.4 Source Code

10.3.5 Hyperlinks

10.3.6 Sections

10.3.7 Explicit Markup

10.3.8 Directives

10.3.9 Footnotes

10.3.10 Comments

Chapter 11

Silence Warnings From the Test Suite

Chapter 12

Silence Warnings From the Test Suite

Chapter 13

Fixing “easy” Issues (and Beyond)

Chapter 14

Issue Tracking

14.1 Using the Issue Tracker

14.1.1 Checking if a bug already exists

14.1.2 Reporting an issue

14.1.3 Understanding the issue's progress and status

14.2 Disagreement With a Resolution on the Issue Tracker

14.3 Helping Triage Issues

14.3.1 Classifying Reports

14.3.2 Reviewing Patches

14.3.3 Finding an Issue You Can Help With

14.4 Gaining the "Developer" Role on the Issue Tracker

14.5 The Meta Tracker

Chapter 15

Triaging an Issue

15.1 Python triage team

15.2 Becoming a member of the Python triage team

15.2.1 GitHub Labels for PRs

15.3 Fields in the Issue Tracker

15.3.1 Title

15.3.2 Type

15.3.3 Stage

15.3.4 Components

15.3.5 Versions

15.3.6 Priority

15.3.7 Keywords

15.3.8 Nosy List

15.3.9 Assigned To

15.3.10 Dependencies

15.3.11 Superseder

15.3.12 Status

15.3.13 Resolution

15.3.14 Mercurial Repository

15.4 Generating Special Links in a Comment

15.5 Checklist for Triaging

Chapter 16

Following R's Development

16.1 Mailing Lists

16.2 Zulip

16.3 IRC

16.4 Blogs

16.5 Standards of behaviour in these communication channels

16.6 Setting Expectations for Open Source Participation

16.7 Additional Repositories

Chapter 17

Porting R to a new platform

Chapter 18

How to Become a Core Developer

18.1 What it Takes

18.2 What it Means

18.3 Gaining Commit Privileges

18.3.1 Mailing Lists

18.3.2 Sign a Contributor Agreement

18.3.3 Pull Request merging

18.4 Responsibilities

Chapter 19

Developer Log

19.1 Procedure for Granting or Dropping Access

Chapter 20

Accepting Pull Requests

20.1 Is the PR ready to be accepted

20.1.1 Does the test suite still pass?

20.1.2 Patch checklist

20.2 Handling Others' Code

20.3 Contributor Licensing Agreements

20.4 Checking if the CLA has been received

20.5 What's New and News Entries

20.6 Working with Git

20.6.1 Active branches

20.6.2 Backporting Changes to an Older Version

20.6.3 Reverting a Merged Pull Request

Chapter 21

Development Cycle

21.1 Branches

21.1.1 In-development (main) branch

21.1.2 Maintenance branches

21.1.3 Security branches

21.1.4 End-of-life branches

21.2 Stages

21.2.1 Pre-alpha

21.2.2 Alpha

21.2.3 Beta

21.2.4 Release Candidate (RC)

21.2.5 Final

21.3 Repository Administration

21.3.1 Organization Repository Policy

21.3.2 Organization Owner Policy

21.3.3 Current Owners

21.3.4 Repository Administrator Role Policy

21.3.5 Current Administrators

21.3.6 Repository Release Manager Role Policy

Chapter 22

Continuous Integration

22.1 Checking results of automatic builds

22.2 Stability

22.3 Flags-dependent failures

22.4 Ordering-dependent failures

22.5 Transient failures

22.6 Custom builders

Chapter 23

Adding to the Stdlib

23.1 Adding to a pre-existing module

23.2 Adding a new module

23.2.1 Acceptable Types of Modules

23.2.2 Requirements

23.2.3 Proposal Process

Chapter 24

Experts Index

24.1 Stdlib

24.2 Tools

24.3 Platforms

24.4 Miscellaneous

24.5 Documentation Translations

Chapter 25

gdb Support

25.1 gdb 7 and later

25.2 gdb 6 and earlier

25.3 Updating auto-load-safe-path to allow test_gdb to run

Chapter 26

Exploring R Internals

Chapter 27

Changing R's Grammar

Chapter 28

Design of R Compiler

Chapter 29

Design of R's Garbage Collector

Chapter 30

Updating standard library extension modules

Chapter 31

Coverity Scan

Chapter 32

Dynamic Analysis with Clang

Chapter 33

Running a buildbot worker

Chapter 34

Core Developer Motivations and Affiliations

34.1 Published entries

34.2 Goals of this page

34.3 Limitations of Scope

Chapter 35

Cheatsheets

Chapter 36

Accepting Pull Requests

36.1 Is the PR ready to be accepted

36.1.1 Does the test suite still pass?

36.1.2 Patch checklist

36.2 Handling Others' Code

36.3 Contributor Licensing Agreements

36.4 Checking if the CLA has been received

36.5 What's New and News Entries

36.6 Working with Git

36.6.1 Active branches

36.6.2 Backporting Changes to an Older Version

36.6.3 Reverting a Merged Pull Request